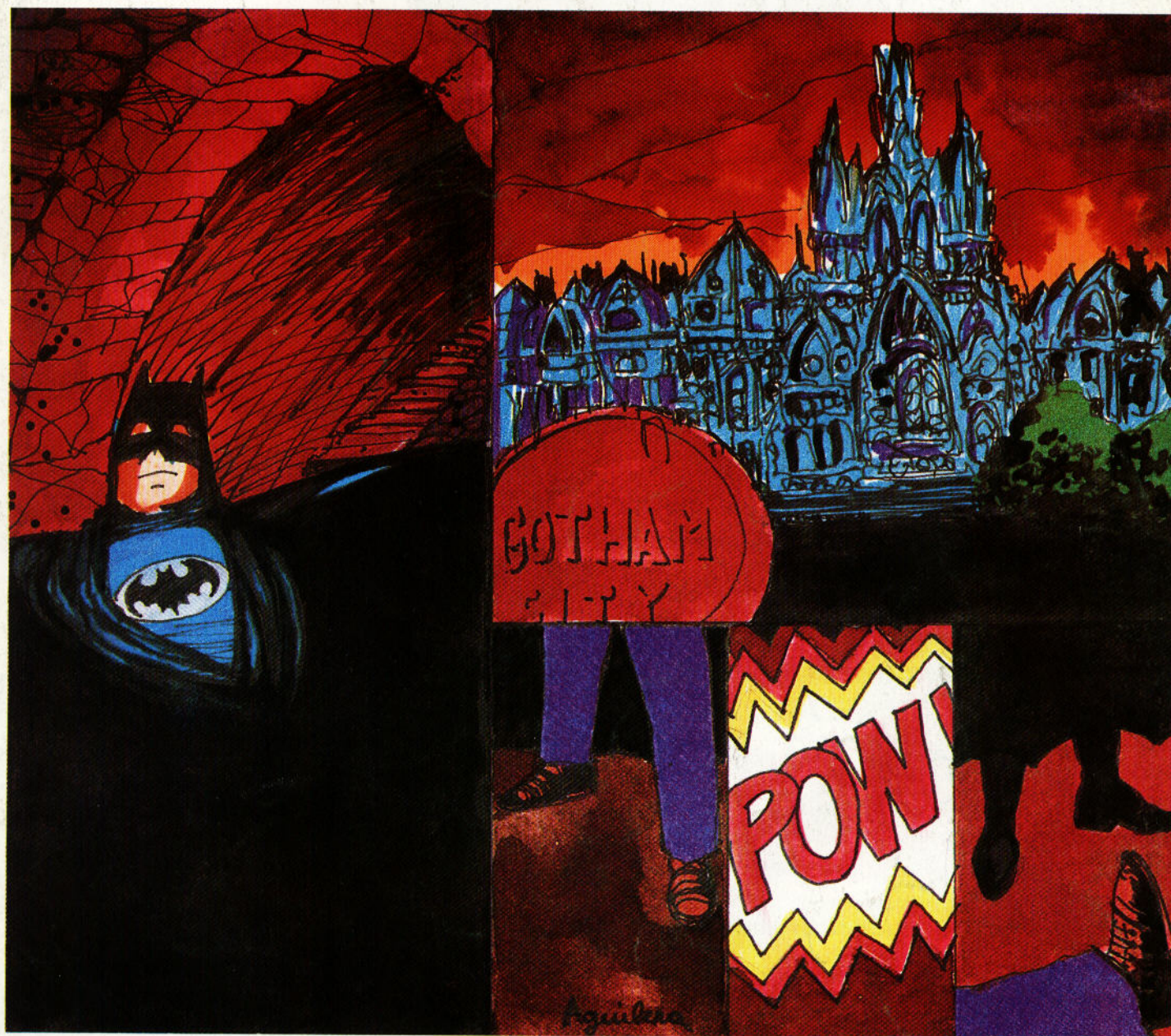
ANO 1
Nº 10
JULHO 1989
NCz$ 4,00

CPU

O BASIC MSX PERSONALIZADO
Novos comandos para o seu Basic

PROGRAMAÇÃO PROFISSIONAL NO MSX
A criação de janelas

BATMAN
O jogo

Software agora tem so

INÉDITO!

MULTICOPY

Enfim o Copiador que você esperava! Realiza cópias Disco/Disco; Disco/Fita; Fita/Fita; Fita/Disco; Disco/Fita automático; Diretório na impressora; Formata e coloca o diretório do seu disco em ordem alfabética, dispondo ainda de mais de sete opções de velocidades para gravação em fita, além de muitos outros recursos.

MSX DESIGNER

Super Editor Gráfico com 40 fontes de letras, saída para impressora em duplo tamanho com escala de cinza (somente em disco).

MSX VÍDEO GRAPHICS PLUS

Sensacional lançamento da Softnew!
A Softnew coloca a disposição dos usuários do MSX, este escelente Editor que irá ajudá-lo na confecção de seus gráficos, com novos formatos e várias outras opções.

JOGOS

A emoção e a aventura esperam por você na Softnew! São mais de 2.000 jogos, além dos mais recentes lançamentos do mercado.
A Softnew também é lazer e entretenimento.

NOVIDADE

Super Snake II — Sensacional jogo, totalmente desenvolvido pela Softnew.

PROGRAMAS

Supercalc II (Compucenter e Princesware) • dBase II Plus (Datalógica e Princesware).

SUPRIMENTOS

Fitas para impressoras • Disquetes • Formulários Contínuos.

PERIFÉRICOS

Monitor para MSX • Drives para MSX 3 1/2 e 5 1/4 • Cartões de 80 colunas para MSX.

ACESSÓRIOS

Table News — Mesa com plano regulável • Box News — Caixa com capacidade para 70 disquetes • Capas protetoras.

LITERATURAS

Programação avançada em MSX • Sistema de disco para MSX • Coleção de programas volume II • Linguagem Basic MSX • Dominando o Expert • Circuitos eletrônicos MSX • Programação profissional em Basic: MSX; IBM-PC; MBASIC • Manual do Drive Leopard 3 1/2.

ESTA É A SUA GRANDE CHANCE!

Se você tem um software criado por você, procure-nos. Nós incrementamos, legalizamos e promovemos o seu software.

É a Softnew em busca de novos talentos na informática.

brenome:

Começou a Maratona Softnew. Uma corrida ao software para o seu MSX.

Se você é usuário do MSX e não quer ficar correndo de loja em loja a procura do software ideal para o seu equipamento, esta é a sua oportunidade. A Softnew coloca a sua disposição toda uma linha de Programas Utilitários/Aplicativos especialmente criada para você tirar o máximo de aproveitamento do seu micro.
Participe da nossa maratona. Na Softnew você sempre ganha, com certeza.

SOS S.A.U.

O cliente Softnew nunca fica a ver navios. Para isso, nós criamos o S.A.U. — Serviço de Atendimento ao Usuário — um sistema tira-dúvidas para auxiliá-lo no que for necessário.
Informe-se. Você não paga nada por este serviço.

TOME NOTA

Se você mora em São Paulo, visite-nos. Seu pedido será atendido na hora.

Pedidos de outras capitais serão entregues em apenas 7 dias.

Todos os programas da Softnew têm garantia total de 180 dias.

SOLICITE CATÁLOGO 89 GRÁTIS

REVENDEDORES SOFTNEW

SÃO PAULO — SP
Filcril (011) 220-3833 • Lima Informática (011) 203-6022 • Pró-Eletrônica (011) 223-6090

SANTOS — SP
Data Market (0132) 35-7500

RIBEIRÃO PRETO — SP
ALS — Comércio de Equipamentos Eletrônicos (016) 636-5379

SÃO BERNARDO DO CAMPO — SP
Microspend (011) 448-6288 (Golden Shopping)

RIO DE JANEIRO — RJ
Inforteles (021) 751-5078 • Teletatch Informática (0242) 52-1483 (Três Rios)

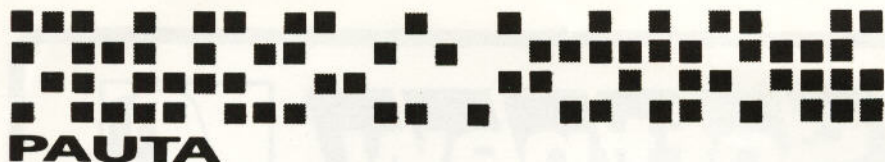
CEARÁ
Top Data Informática (085) 239-1618 • Sun Photo Ltda. (085) 244-2308

BRASÍLIA — DF
Hall Informática (061) 248-4755

PORTO ALEGRE — RS
Prólogos Informática (0512) 22-5803

SOFTNEW INFORMÁTICA LTDA.
Rua Miguel Maldonado, 173 — Jd. São Bento
CEP 02524 — São Paulo — SP
Tel.: (011) 266-2902

FSILVA

PAUTA

ÁGUIA INFORMÁTICA LTDA.
AV. N. S. DE COPACABANA, 605/804
COPACABANA
22040 – RIO DE JANEIRO – RJ
TELEFONE: 021-235.3541

DIRETOR RESPONSÁVEL
GONÇALO R. F. MURTEIRA

DIRETOR COMERCIAL
JOSÉ GERALDO J. NETO

DIRETOR ADMINISTRATIVO
JOSÉ IDEMAR A. NASCIMENTO

ASSESSORIA TÉCNICA
DIVINO C. R. LEITÃO

JORNALISTA RESPONSÁVEL
DOLAR TANUS
REGISTRO 430-RS

COLABORADORES
PEDRO HENRIQUE GAMA
PAULO MARQUES FIGUEIRA
SÉRGIO GUY PINHEIRO ELIAS
PAULO ROBERTO PINHEIRO ELIAS
BRUNO MARRUT
JÚLIO VELLOSO
SÉRGIO DURIC CALHEIROS
DIVINO LEITÃO
GUILHERME A. L. DA SILVA
ANDRÉ L. A. SANTOS
MARCOS R. TAVARES
EDUARDO R. TAVARES

REVISÃO DE TEXTO
LAURA MARIA PINTO

CAPA
JOSÉ AGUILERA

IMPRESSÃO
EDITORA LORD

DISTRIBUIÇÃO
FERNANDO CHINAGLIA DISTRIBUIDORA

NOVO EDITOR GRÁFICO

A RIOSOFT e a NEMESIS lançam o mais novo editor gráfico para a linha MSX, o EASYgraph.

Este novo produto é composto por 4 módulos independentes.

O 1º é um editor de símbolos vetores. Com ele pode-se criar alfabetos e/ou símbolos para posterior uso no módulo 4.

O 2º é um programa capaz de fazer apresentações, aberturas para vídeos cassetes, propaganda eletrônica. No disco mestre existem dois arquivos. Um mostra as propriedades do EASYgraph e o outro é um tutor que ensina como utilizar o programa. Ambos foram criados pelo próprio EASY no módulo 4.

O 3º módulo serve para imprimir telas em impressora gráfica, com variações de cinza.

O 4º e último módulo é um poderoso editor gráfico com mais de 40 funções, todas acessadas pelo simples toque da tecla correspondente.

Entre as funções disponíveis, temos: arcos, linha, raio, círculos, elipses, retângulos, polígonos regulares quaisquer, apagamento por área ou com borracha, variação da largura do traço, variação da largura do traço e da velocidade do cursor, pintura com pincel, spray ou "fill", e a sua principal característica, que é o uso de caracteres vetoriais para textos e símbolos.

Com este tipo de caracter suprime-se uma limitação que existia com o uso de caracteres matriciais, que era o tamanho e a inclinação do texto, que agora podem ser definidos na hora do seu uso, sem necessidade de alterar a matriz já elaborada no 1º módulo.

Esta nova opção para os usuários é 100% nacional e de autoria do Eng. Civil Afonso C. C. Pragana.

NEWSOFT

O pessoal que trabalha em computação gráfica no MSX ou que dele se utiliza para produzir aberturas em vídeos, está dando pulos de alegria.

A Newsoft Informática está comercializando um programa denominado E.V.A. – Editor de Vinhetas Animadas, que possibilita ao usuário criar os mais espetaculares efeitos visuais, promovendo um verdadeiro show à parte em suas telas de abertura.

O programa permite que as vinhetas criadas por este editor sejam movimentadas em todas as direções, e que as letras ou desenhos promovam giros de até 360 graus.

Tais efeitos podem ser utilizados de forma isolada ou em conjunto, com telas previamente produzidas por qualquer editor gráfico existente no mercado ou em telas em basic.

O programa, além de possuir todo um suporte técnico, está sendo comercializado com um vastíssimo manual, por apenas NCz$ 50,00 (disco incluído).

Os interessados poderão efetuar seus pedidos diretamente à Newsoft Informática – Av. Nilo Peçanha, 50 – sala 906 – Rio de Janeiro – RJ – CEP 20020.

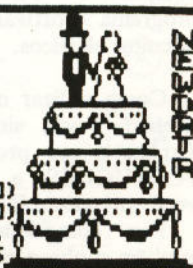
PAULISOFT

A Paulisoft está de instalações novas, ocupando agora um conjunto de 3 salas, localizadas a poucos metros da estação do metrô Anhangabaú, onde poderá ser encontrado tudo para MSX, desde a MEGARAM aos mais variados jogos e aplicativos.

O novo endereço é: Rua Coronel Xavier de Toledo, 1231 – Conjuntos 31/32 – São Paulo – SP – CEP 01048.

MSX 2

Já está sendo vendido pela MPO o kit que transforma seu MSX 1 em MSX 2.0.

Este kit, que pode ser comprado no Mappin, Breno rossi, entre outras, é instalado em uma hora, pela própria MPO, em sua oficina, na Av. Paulista, por técnicos especializados.

O BASIC MSX PERSONALIZADO

LUIZ CARLOS BITTENCOURT

A informática, desde sua origem, sempre está "acompanhada" por problemas de duas naturezas principais:

1) A existência de equipamentos capazes de armazenar, processar e transferir dados com grande capacidade e velocidade a custo acessível.

2) A facilidade de utilização destes equipamentos pelos usuários.

A primeira classe de problemas relacionada aos equipamentos (Hardware), tem apresentado processos significativos, com a utilização de tecnologias cada vez mais avançadas, tais como meios magnéticos/óticos de armazenamento/comunicações de dados e processadores de alta tecnologia.

Os custos destes equipamentos já possibilitam sua utilização em grande escala.

A segunda classe de problemas envolve uma questão mais sutil: como utilizar os recursos de que a máquina dispõe?

Imagine que você tem um problema: descobrir os tipos de fios elétricos adequados para colocar em sua nova casa.

Imagine, também, duas diferentes alternativas para a solução do seu problema:

1) Você conta com a assessoria de um engenheiro especialista em circuitos elétricos.

2) Você possui um microcomputador equipado e "Diskette" contendo um programa (Software) para cálculos de circuitos elétricos.

Como utilizar os serviços do engenheiro? Você, simplesmente, FALA! Explica o seu problema, utilizando o PORTUGUÊS, a LINGUAGEM que você aprendeu e fala desde criança.

Por coincidência, o engenheiro também conhece muito bem esta linguagem, e vai "entender" a sua necessidade e utilizar os conhecimentos que possui para ajudá-lo.

As dúvidas mútuas serão resolvidas por recursos da própria linguagem de comunicação.

E como utilizar os "serviços" do microcomputador?

Eis a QUESTÃO-CHAVE!

O computador AINDA não é capaz de se comunicar com o usuário na sua própria linguagem. O que acontece é exatamente o contrário: o usuário é que tem que aprender a linguagem do computador (Diskette, Teclado, Tela, Menus, etc).

Como ligar a máquina? O que fazer com o "Diskette"? Que teclas apertar? O que responder?

Na raiz destes problemas está a LINGUAGEM DE COMUNICAÇÃO entre o computador e o usuário, questão, portanto, de fundamental importância para que os recursos da máquina possam ser completamente utilizados.

No caso do MSX, uma linguagem de comunicação já foi armazenada em "memória permanente" (ROM).

Este software, o INTERPRETADOR BASIC-MSX, "converte" a linguagem BASIC-MSX (mais "amigável" e próxima da linguagem do usuário) em "linguagem de máquina" (códigos binários que podem ser entendidos pelo Z-80, o processador principal do MSX).

O BASIC-MSX possibilita a utilização dos recursos da máquina de maneira simplificada, bastando, para isto, ligar o micro.

Apesar da Linguagem BASIC-MSX ser de fácil entendimento, como tudo na vida, possui seus defeitos.

Este artigo se propõe a transformar estes defeitos em virtudes, oferecendo algumas técnicas com as quais você poderá modificar/complementar as funções do Interpretador BASIC (que, obviamente, é implementado na "linguagem de máquina do Z-80), de acordo com as suas próprias necessidades.

ADAPTANDO AS FUNÇÕES DO INTERPRETADOR-BASIC

O INTERPRETADOR-BASIC pode executar comandos do usuário de duas maneiras:

1) No modo DIRETO você escreve uma linha de comandos e ela é imediatamente executada. Este modo é identificado pela ausência de dígito numérico no início da linha.

2) No modo PROGRAMA, uma seqüência de linhas de comandos previamente armazenados em uma memória RAM (modificável) é executada quando o usuário emprega o comando RUN (Execute). Estas linhas devem ter sido previamente digitadas, iniciando por números que as identificam.

ALGUMAS TÉCNICAS PARA ADPTAR O INTERPRETADOR BASIC

1) Utilizar as funções DEFURS/USR para acionar "rotinas" em linguagem Z-80.

2) Utilizar os GANCHOS (HOOKS) para "desviar" as rotinas normais do Interpretador, modificando-as.

3) Utilizar o comando CALL para acionar rotinas em linguagem de máquina.

4) "Desviar" a rotina encarregada de "Receber uma Linha Digitada pelo Usuário", tratando esta linha ANTES que ela seja processada pelo BASIC.

Esta técnica possibilita a modificação da sintaxe dos próprios comandos BASIC, além de introduzir novos comandos (o software/livro BIT-BASIC dela se utiliza e a descreve detalhadamente).

5) Implementar funções para os comandos CMD e IPL, que são reconhecidos pelo interpretador Basic, possuem um GANCHO próprio, porém geram apenas uma mensagem de erro quando são utilizados.

A seguir mostraremos uma rotina implementada a partir do comando CMD, destinada a faciltar a utilização do MSX a partir do BASIC.

Ela inclui novas "palavras" à linguagem original, destinadas a tornar mais fácil a movimentação de dados de diversos formatos DE/PARA o BUFFER de dados de um arquivo (FIELD), dispensando o uso de comandos MKI$, MKS$, CVI, CVS e CVD, de utilização bastante trabalhosa.

ROTINA PARA "AGRUPAR/DESAGRUPAR" VARIÁVEIS BASIC

A rotina a seguir, implementada a partir do comando CMD, destina-se a modificar a linguagem BASIC utilizada para transferir dados de diversos formatos para o BUFFER de dados de um arquivo.

Analise o seguinte programa BASIC.

figura 1

```
10 MAXFILES=1
20 OPEN"A:TESTE"AS#1
30 FIELD #1,2ASN1$,4ASN2$,
8ASN3$,20ASN4$
40 A%=1:B!=2:C#=3:D$="ABC"
50 RSETN1$=MKI$(A%):RSETN2
$=MKS$(B!):RSETN3$=MKD$(C#
):LSETN4$=D$
60 PUT#1,1
70 A%=0:B!=0:C#=0:D$=""
80 PRINTA%;B!;C#;D$
90 GET#1,1
100 A%=CVI(N1$):B!=CVS(N2$
):C#=CVD(N3$):D$=N4$
110 PRINTA%;B!;C#;D$
120 CLOSE#1
130 END
```

A SINTAXE (Regras de escrita) normal do BASIC prevê a definição dos campos de dados de um registro pelo comando FIELD, e a movimentação dos dados de/para estes campos pelos comandos MKI$, MKS$, MKD$, CVI, CVS e CVD.

O BASIC faz com que as variáveis definidas no comando FIELD "apontem" para endereços pertencentes ao "BUFFER do arquivo, ou seja, quando colocamos dados nestes campos, eles são diretamente armazenados no registro/BUFFER.

Após a implementação da rotina para "agrupar/desagrupar variáveis", este mesmo programa poderá ser escrito da seguinte maneira:

figura 2

```
10 MAXFILES=1
20 OPEN"A:TESTE"AS#1
30 FIELD#1,34ASX1$
40 A%=1:B!=2:C#=3:D$="ABC"
50 CMDPUTX1$,A%,B!,C#,D$
60 PUT#1,1
70 A%=0:B!=0:C#=0:D$=""
80 PRINTA%;B!;C#;D$
90 GET#1,1
100 CMDGETX1$,A%,B!,C#,D$
110 PRINTA%;B!;C#;D$
120 CLOSE#1
130 END
```

A codificação do comando FIELD fica simplificada, bastando definir apenas uma única variável do tipo STRING de tamanho igual ao tamanho do registro.

A colocação dos dados seqüêncialmente no registro, mantendo seus formatos originais (Variáveis Inteiras, de Precisão Simples, de Precisão Dupla, String), passa a ser feita com a sintaxe mostrada nas linhas 50 e 100, colocando, simplesmente, a variável definida no comando FIELD (Variável "Grupo"), seguida das variáveis a serem movimentadas (Variáveis "Individuais"), separadas por vírgulas.

Os "Novos Comandos BASIC" "CMDGET" e "CMDPUT" se encarregarão de "entender" a nova sintaxe e executar os procedimentos descritos.

A codificação do programa BASIC fica muito simplificada, especialmente quando trabalhamos com um grande número de arquivos/campos.

Além disso, o programa fica mais "limpo", facilitando a sua análise, importante fator de qualidade de programação.

CARACTERÍSTICAS DA ROTINA IMPLEMENTADA

1) Implementa as novas funções a partir do comando CMD, já previsto pelo BASIC, mas que apenas apresenta a mensagem "FUNÇÃO ILEGAL", quando utilizado.

2) Se o tamanho da variável "Grupo" (FIELD) for maior ou menor que a soma dos tamanhos das "Variáveis Individuais" (em Bytes), será gerado um erro de "FUNÇÃO ILEGAL".

3) A Rotina é instalada nas posições de memória correspondentes à "Fila Musical C", do comando PLAY.

ROTINAS DO BASIC UTILIZADAS

#SEA4 – BUSCA VARIÁVEL

No início, o par de registradores HL aponta para o INÍCIO DO NOME DA VARIÁVEL, no TEXTO DO PROGRAMA.

No final, HL aponta para a posição imediatamente seguinte ao nome da variável, e o par de registradores DE aponta para o primeiro BYTE do CONTEÚDO DA VARIÁVEL na ÁREA DE ARMAZENAMENTO DE VARIÁVEIS.

EXPLICAÇÃO DETALHADA DA ROTINA

Neste ponto, é necessário explicar a seqüência de procedimentos do BASIC para tratamento do comando CMD.

Quando você digita uma linha modo "PROGRAMA" (iniciando por dígitos numéricos), os comandos BASIC nela contidos são "Codificados" ou "Tokenizados" (convertidos em TOKEN = SINAL). Por exemplo, o comando PRINT é convertido para #91 (&H91), o comando CMD é convertido para #D7, o comando GET para #B2 e o comando PUT para #B3.

O comando CMDEGT, portanto, estará guardado como "#D7 #b2, e o comando CMPDUT como "#D7 #B3".

Quando você executa o programa com "RUN", o BASIC utiliza os registradores HL para "percorrer" as linhas de programa BASIC (BYTE a BYTE), processando um por um os comandos por você escritos. Quando é encontrado um BYTE contendo #D7, o Interpretador BASIC reconhece o comando CMD e passa a executar esta função.

A rotina CMD original, instalada a partir de #7C34, faz uma "chamada" (CALL) para o GANCHO em #FE0D e, em seguida, desvia para a rotina de erro que apresenta a mensagem "FUNÇÃO ILEGAL".

A nova rotina CMD, a seguir descrita, intercepta o GANCHO em #FE0D e implementa as novas funções já citadas.

```
 10 CMDVAR:  ORG   #FA75
 20 ; TIPO COMANDO
 30          POP   BC
 40          LD    A,(HL)
 50          SUB   #B2
 60          JR    C,ERRO2
 70          CP    #02
 80          JR    NC,ERRO2
 90          XOR   #01
100          LD    (VAR1+1),A
110          LD    (VAR3+1),A
120          RST   #10
130          JR    Z,ERRO2
140 ; VARIAVEL "GRUPO"
150          CALL  ENDVAR
160          JR    Z,ERRO2
170          PUSH  DE
180          PUSH  BC
190 ; VARIAVEIS "UNITARIAS"
200 PRXVAR:  LD    A,(HL)
210          CP    #3A
220          JR    Z,RETORNA
230          CP    #00
240          JR    Z,RETORNA
250          CP    #2C
260          JR    NZ,ERRO1
270          RST   #10
280          JR    Z,RETORNA
290          CALL  ENDVAR
300          JR    Z,ERRO1
310 ; VERIFICA TAMANHO
320          EX    (SP),HL
330          AND   A
340          SBC   HL,BC
350          JR    C,ERRO1
360          EX    (SP),HL
370 ; MOVE VAR.UNITARIA DE/PARA VAR.GRUPO
380          POP   IX
390          EX    (SP),HL
400 VAR1:    JR    VAR2
410 VAR2:    EX    DE,HL
420          LDIR
430 VAR3:    JR    VAR4
440 VAR4:    EX    DE,HL
450          EX    (SP),HL
460          PUSH  IX
470          JR    PRXVAR

480 ; OBTEM ENDEREÇO VARIAVEL
490 ENDVAR:  CALL  #5EA4
500          LD    A,(#F663)
510          CP    #03
520          JR    NZ,VAR6
530          LD    A,(DE)
540          INC   DE
550          LD    (VAR5+2),DE
560 VAR5:    LD    DE,(#0000)
570          CP    #00
580 VAR6:    LD    C,A
590          LD    B,#00
600          RET
610 ; ROTINA ERRO
620 ERRO1:   POP   BC
630          POP   BC
640 ERRO2:   JP    #7C37
650 ; FIM VARIAVEIS
660 RETORNA: POP   BC
670          POP   DE
680          LD    A,B
690          OR    C
700          JP    NZ,#7C37
710          RET
720          ***
```

*** Instrução 10

Instrui o programa. Compilador Assembler Z-80 para colocar o Programa em linguagem de máquina (Programa Objeto) a partir do endereço #FA75, correspondente ao início da área reservada para a "Fila Musical C", do comando PLAY.

*** Instruções 30 a 130

Verificam o "Tipo de Comando" que está sendo processado (CMDGET ou CMDPUT).

*** Instrução 30

Retira da PILHA DO SISTEMA o endereço de retorno à chamada do GANCHO em #FE0D pela rotina CMD.

A posição seguinte da PILHA contém o endereço de retorno normal ao Interpretador BASIC. Se fosse executada uma instrução RET do Z-80, o programa continuaria normalmente sua execução.

Se o GANCHO não tivesse sido interceptado, seria efetuado um retorno para a rotina CMD, que faria um desvio para #7C37, apresentando a mensagem "FUNÇÃO ILEGAL".

*** Instrução 40

Carrega no Registrador A o conteúdo do BYTE apontado pelos Registradores HL, que apontam para o BYTE imediatamente seguinte ao comando CMD (#D7), o qual deve conter #B2 ou #B3 (GET OU PUT).

*** Instrução 50

Subtrai #B2 do Registrador A

*** Instrução 60

Se a substituição anterior resulta "menor que zero" (Reg-A continha valor menor que # B2, ou seja, nem GET nem PUT), desvia para a rotina ERRO2 para apresentação da mensagem FUNÇÃO ILEGAL.

*** Instrução 70

Compara REG-A com #02

*** Instrução 80

Se Reg-A não contiver #00 ou #01 (correspondentes aos valores originais #B3 ou #B4), desvia para ERRO2.

*** Instrução 90

Efetua uma operação XOR (OR EXCLUSIVO) sobre Reg-A, com o valor # 01.

Se Reg-A continha #00, passa a conter #01, e, se continha #01 passa a conter #00 após esta instrução.

*** Instrução 100

Coloca o valor de Reg-A no campo endereço da instrução VAR1 (Instrução 400 – Veja explicações nesta instrução).

*** Instrução 110

Idem, no campo de endereço da instrução VAR3.

*** Instrução 120

Efetua uma chamada ao endereço #0010, que executa a rotina CHRGTR do BIOS (Basic Input Output System – Sistema Básico de Entrada e Saída), encarregada de "Ler" o próximo caracter apontado pelos registradores HL, normalmente utilizada para "Ler o próximo caracter do TEXTO DO PROGRAMA BASIC".

Esta rotina "Soma 1" em Reg-HL e coloca em Reg-A o byte apontado por Reg-HL, além de posicionar os Indicadores de Status C e Z para indicar se este BYTE é numérico ou se é #00.

*** Instrução 130

Se o BYTE contido em Reg-A contiver #00 90 (fim da linha de programa foi encontrado), desvia para ERRO2.

Caso contrário, este deve ser o primeiro digito correspondente ao "Nome da Variável definida no comando FIELD". E esta a sintaxe definida para os comandos CMDGET e CMDPUT, lembra-se?

*** Instruções 150 a 180

Obtém o endereço de memória apontado pela Variável String "Grupo".

*** Instrução 150

Chama (CALL) a rotina ENDVAR (veja instruções 490 a 600), que aciona a rotina do BASIC encarregada de obter o endereço da variável.

*** Instrução 160

Caso o Indicador Z "retorne ligado" daquela rotina, desvia para ERRO2.

*** Instrução 170

Armazena na PILHA o conteúdo do Reg-DE, onde foi colocado o "Endereço da Variável" pela rotina ENDVAR.

*** Instrução 180

Armazena na PILHA o conteúdo do Reg-BC, onde foi colocado o "Tamanho da Variável" (em Bytes).

*** Instruções 200 a 300

Obtém o endereço dos dados da "próxima variável", separada da variável anterior por uma vírgula.

*** Instrução 200

Coloca em Reg-A o Byte apontado por Reg-HL (BYTE imediatamente após o nome da variável anterior).

*** Instrução 210

Compara REG-a com #3A (dois pontos).

*** Instrução 220

Se Reg-A contém ":", desvia para a rotina RETORNA, encarregada de encerrar esta rotina para o comando CMD, já que o caracter ":" é utilizado para "encerrar" cada comando BASIC.

*** Instrução 230

Compara Reg-A com #00.

*** Instrução 240

Se Reg-A contém #00 (o fim da linha de comandos BASIC foi alcançado), desvia para a rotina RETORNA.

*** Instrução 250

Compara Reg-A com #2C (vírgula).

*** Instrução 260

Se Reg-A não contém uma vírgula (#2C), desvia para a rotina ERRO1. Portanto, a sintaxe dos comandos CMDGET/CMDPUT exige a separação com vírgulas dos nomes das varíaveis.

*** Instrução 270

Obtém em Reg-A o próximo BYTE do texto do programa BASIC.

*** Instrução 280

Caso o fim da linha de comandos tenha sido alcançada, desvia para a instrução RETORNA, com fim "normal" da rotina.

*** Instrução 290

Chama a rotina ENDVAR, que obtém o endereço/tamanho da variável.

*** Instrução 300

Se indicador-Z ligado, desvia para ERRO1.

*** Instruções 320 a 360

Verifica se os BYTES da Variável Grupo que ainda restam são suficientes para comportar esta Variável Individual.

*** Instrução 320

"Troca" o conteúdo de Reg-HL com o conteúdo da última posição da PILHA, onde foi guardado o "Tamanho da Variável Grupo" (veja instrução 180).

*** Instrução 330

Executa uma operação AND de Reg-A sobre ele próprio, objetivando colocar "zero" no Indicador-C (Carry-Status).

*** Instrução 340

Subtrai de Reg-HL (Tamanho da Variável Grupo) o valor contido em Reg-BC (Tamanho da Variável Individual).

A instrução SBC (Subtract With Carry) subtrai também de Reg-HL o valor do Indicador-C, por isto ele foi "zerado" na instrução 330.

*** Instrução 350

Caso a subtração da instrução 340 resulte "menor que zero" o indicador C é ligado. Neste caso, desvia para ERRO1.

*** Instrução 360

"Troca o conteúdo de Reg-HL (Número de Bytes que ainda sobram na Variável Grupo) com a PILHA (posição do INTERPRETADOR BASIC na linha de texto do programa, aí colocada pela instrução 320).

*** Instrução 380 a 470

Movem os BYTES da Variável Individual para a Variável Grupo, ou vice-versa, de acordo com o tipo de comando (CMDPUT ou CMDGET).

*** Instrução 380

"Salva" em Reg-IX a última posição da PILHA.

*** Instrução 390

Troca o conteúdo de Reg-HL (Posição do Texto) com o conteúdo da PILHA (Endereço da Variável Grupo, aí colocado pela instrução 170).

*** Instrução 400

Desvia para a instrução 410 (VAR2) ou para a instrução 420, conforme o seu operando de endereço seja "0" ou "1", valor aí colocado pelas instruções 40 a 110, de acordo com o tipo de comando.

*** Instrução 410

Somente executada se o comando é CMDPUT.

Troca os conteúdos de Reg-DE e Reg-HL.

*** Instrução 420

Move a quantidade de BYTES contida em Reg-BC (tamanho da Variável Individual), do endereço apontado por Reg-HL, que pode ser um endereço da Variável Grupo ou o endereço da Variável Individual, conforme a instrução 410 tenha ou não sido executada, para o endereço apontado por Reg-DE (Idem).

No final, Reg-HL e Reg-DE são aumentados do valor contido em Reg-BC, e este é zerado.

*** Instruções 430 a 440

Utilizando a mesma técnica das instruções 400 e 410, troca ou não os conteúdos de Reg-HL e Reg-DE.

*** Instrução 430

Desvia para a instrução 440 (VAR4) ou para a instrução 450.

*** Instrução 440

Troca os conteúdos de Reg-HL e Reg-DE.

*** Instrução 450

Troca o conteúdo de Reg-HL (próximo endereço na Variável Grupo, resultante da instrução 420 = LDIR), com o conteúdo da PILHA (endereço do próximo BYTE no texto do programa).

*** Instrução 460

Coloca na PILHA o "tamanho restante" da variável Grupo, salvo em Reg-IX pela instrução 380.

*** Instrução 470

Desvia para a instrução 200 (PRXVAR) para dar tratamento à próxima Variável Individual.

*** Instrução 490 a 600

Estas instruções utilizam a rotina #5EA4 do BASIC para obter o endereço/tamanho de cada variável.

*** Instrução 490

Efetua chamada (CALL) à rotina do BASIC que inicia no endereço #5EA4, destinada a efetuar a "Busca de Variável".

No início, Reg-HL deve apontar para o primeiro BYTE do "Nome da Variável" na linha de texto do programa.

No final, Reg-HL aponta para o caracter imediatamente seguinte ao nome da variável, Reg-DE aponta para o primeiro BYTE do "Conteúdo da Variável", na área de armazenamento de variáveis.

E conveniente lembrar que, para variáveis do tipo STRING, Reg-DE não aponta diretamente para o "conteúdo" da variável. Veja descrição em seguida.

Armazenamento de Variáveis

xx yy yy zz . . . zz

xx = tipo de variável
02 = Inteira
03 = String
04 = Simples
08 = Dupla

yy yy = Nome da variável

zz . . . zz = Conteúdo da Variável, de acordo com o seu tipo.

tipo 02: dois BYTES, contendo valor numérico entre -32768 e +32767.

tipo 03: três BYTES, o primeiro com o tamanho e os dois seguintes com o endereço do conteúdo da variável.

tipo 04: quatro BYTES, representando um valor numérico de até 6 dígitos de precisão.

tipo 08: oito BYTES, representando um valor numérico de até 14 dígitos de precisão.

*** Instrução 500

Carrega em Reg-A o campo "VALTYP" (tipo de variável) do BASIC, armazenando no endereço #F663, onde a rotina #5EA4 coloca o tipo de Variável.

*** Instrução 510

Compara Reg-A com #03

*** Instrução 520

Se a variável não é o tipo 03 (String), desvia para a instrução 580 (VAR6).

*** Instrução 530

Se a variável é do tipo String, carrega em Reg-A o BYTE apontado por Reg-DE, que contém o tamanho desta variável em BYTES.

*** Instrução 540

"Soma um" em Reg-DE, que passa a apontar para os dois BYTES que contêm o endereço da variável String.

*** Instrução 550

Carrega no campo de endereço da instrução 560 (VAR5) o valor contido em Reg-DE.

*** Instrução 560

Carrega em Reg-De o conteúdo dos BYTES apontados pelo próprio Reg-DE, que é o endereço da variável String.

*** Instrução 570

Compara Reg-A (tamanho da variável String) com zero.

Em caso afirmativo, o indicador-Z, será "ligado", e as instruções 160 e 300 farão o teste desta condição e desviarão para a rotina ERRO.

Portanto, variáveis String "vazias" não serão aceitas pela rotina.

*** Instrução 580

Coloca o valor de Reg-A em Reg-C.

*** Instrução 590

Carrega #00 (zero) em Reg-B.

As instruções 580/590 colocam em Reg-BC o tamanho da variável.

*** Instrução 600

Retorna à instrução imediatamente seguinte àquela que efetuou à rotina ENDVAR.

*** Instrução 620 a 640

Encerra a rotina com mensagem de erro "FUNÇÃO ILEGAL".

*** Instrução 620

Retira da PILHA o tamanho da variável Grupo.

(Você não pode esquecer "pacote" na PILHA).

*** Instrução 630

Retira da PILHA o endereço da variável Grupo.

Programa BASIC para criar rotina VARIAVEIS

```
10 'PARA CRIAR/SALVAR
20 DATAC1,7E,D6,B2,38,5E,FE,02
30 DATA30,5A,EE,01,32,B1,FA,32
40 DATAB6,FA,D7,28,4F,CD,BD,FA
50 DATA28,4A,D5,C5,7E,FE,3A,28
60 DATA46,FE,00,28,42,FE,2C,20
70 DATA39,D7,28,3B,CD,BD,FA,28
80 DATA31,E3,A7,ED,42,38,2B,E3
90 DATADD,E1,E3,18,00,EB,ED,B0
100 DATA18,00,EB,E3,DD,E5,18,D4
110 DATACD,A4,5E,3A,63,F6,FE,03
120 DATA20,0C,1A,13,ED,53,CF,FA
130 DATAED,5B,00,00,FE,00,4F,06
140 DATA00,C9,C1,C1,C3,37,7C,C1
150 DATAD1,78,B1,C2,37,7C,C9,3E
160 DATA00,D3,A8,F1,C9
170 FORI=&HFA75TO&HFAE9:READA$:POKEI,
    VAL("&H0"+A$):NEXTI
180 BSAVE"CMDVAR.EXE",&HFA75,&HFAE9:END
190 'PARA CARREGAR/ATIVAR
200 BLOAD"CMDVAR.EXE
210 POKE(&HFE0D),&HC3:POKE(&HFE0E),&H75:
    POKE(&HFE0F),&HFA
```

*** Instrução 640

Desvia para o endereço #7C37 que fará a apresentação da mensagem de erro "FUNÇÃO ILEGAL".

*** Instrução 660 a 710

Verificar se a Variável Grupo foi totalmente preenchida com Variáveis Individuais.

Em caso afirmativo, devolve o controle ao interpretador BASIC em condição normal.

Se a soma dos tamanhos em BYTES das variáveis individuais não for igual ao tamanho em BYTES da variável grupo, apresenta mensagem de erro.

*** Instrução 660

Retira da PILHA o Tamanho da Variável Grupo, já deduzido dos tamanhos das Variáveis Individuais.

*** Instrução 670

Retira da PILHA o endereço da Variável Grupo.

*** Instrução 680

Coloca em Reg-A o conteúdo de Reg-B.

*** Instrução 690

Compara Reg-A (Reg-B) com Reg-C, BIT a BIT, na modalidade OR.

Esta operação resulta em zero (Indicador-Z ligado) somente se ambos os registradores contiverem zero.

Com as instruções 680/690 estamos verificando se Reg-BC contém ou não "zero".

*** Instrução 700

Se Reg-BC não contém zero, desvia para #7C37 para apresentar mensagem de erro (a variável Grupo não foi totalmente preenchida).

*** Instrução 710

Efetua o retorno ao Interpretador BASIC em situação normal, para que este de continuidade ao processamento dos demais comandos do seu PROGRAMA BASIC.

ASSINE CPU

FAÇA SUA ASSINATURA ANUAL DE CPU E RECEBA UM DISCO CHEIO DE PROGRAMAS, DICAS E JOGOS

ASSINE CPU

Desejo efetuar a assinatura da revista CPU. Para tal, estou enviando cheque nominal à Águia Informática, ou Vale Postal (pagável na Agência Copacabana) no valor de:

( ) NCz$ 19,20 Por assinatura no período de 6 meses.
( ) NCz$ 38,40 Por assinatura no período de 1 ano.

NOME: ____________________
ENDEREÇO: ____________ **TEL.:** ________
BAIRRO: ____________ **CIDADE:** ________
ESTADO: ____________ **CEP:** ________
DADOS DO EQUIPAMENTO: ____________________

* **Promoção válida até 05/08/89 e somente para assinaturas anuais**
* **Despesas de correio por conta da revista**

APPLE VERSUS MSX

Pierluigi Piazzi

O mercado de microcomputadores no Brasil é extremamente peculiar: a estranha lei de informática que o rege lhe conferiu características mais estranhas ainda. A concorrência não é feita entre fabricantes, mas entre as cópias que eles fabricam.

Existem dezenas de fabricantes de clones (eufemismo empregado para denominar cópias descaradamente pirateadas) de Apple e IBM-PC, enquanto apenas dois se aventuram no MSX.

Apesar disso, o volume de MSX vendido nos últimos dois anos foi tão espantoso que chegou a incomodar consideravelmente todos os outros fabricantes. Formou-se um sutil lobby para sabotar o MSX, cujos efeitos se fazem sentir, por exemplo, em estranhos entraves na SEI.

Na qualidade de editor técnico de livros de computação, vou me permitir fazer uma comparação imparcial destas máquinas, no intuito de orientar os usuários, vítimas de vendedores mal intencionados e informações distorcidas. Esta imparcialidade é garantida pelo fato de que publico livros para todas estas máquinas e, conseqüentemente, não tenho interesse em privilegiar uma ou outra.

Inicialmente, me permito excluir os clones do IBM-PC desta comparação, pois a faixa de preço os coloca numa categoria completamente diferente da do Apple e MSX. Vale a pena observar, porém, que vendedores e "assessores" têm tido a tendência a "empurrar" um caro IBM-PC a usuários cujas necessidades poderiam, tranqüilamente, ser preenchidas por máquinas de menor porte e, conseqüentemente, menor custo (neste caso, porém, a comissão seria consideravelmente menor!).

Vou dividir esta comparação nos itens que considero mais significativos:

ARQUITETURA

O MSX é uma máquina de concepção muito mais moderna que o Apple. A rigor, a concepção do MSX é muito mais moderna que a do próprio IBM-PC. Conseqüentemente, muitos erros e falhas do Apple foram corrigidas quando o MSX foi projetado. O MSX não consome memória RAM para criar telas, pois tem uma VRAM (Video RAM) de 16 Kb exclusiva para seu VDP. A ausência de um circuito análogo ao VDP no Apple, a nível de cores, sprites e versatilidade superam os do Apple sem termo de comparação.

Além disso, a existência do PSG, dispositivo de som programável, faz o MSX ganhar de longe na área de produção de sons e música. Este item pode parecer irrelevante (e, a rigor, não é!) quando pensamos em aplicações exclusivamente empresariais, mas torna-se fundamental quando pensamos em aplicações domésticas e educacionais.

Só para exemplificar, em nossa editora editamos um Curso de Música para MSX, para ser utilizado em conservatórios e escolas, que permite, utilizando apenas os recursos de máquina e sem exigir a compra de nenhum "software específico", dar um curso de iniciação musical extremamente eficiente e que não exige a psicomotricidade e apuro técnico que se exigiria no caso de um instrumento musical tradicional adequado para esta tarefa: é o MSX, graças ao PSG citado.

Com relação ao microprocessador principal, estamos comparando o Z80 do MSX com o 6502 do Apple II+ (ou 65C02 do IIe). A maioria dos programas que utilizam Linguagem de Máquina no Brasil concorda em afirmar que o Z80 tem muito mais recursos e instruções que o 6502.

VELOCIDADE DE PROCESSAMENTO

O MSX tem um sistema operacional extremamente versátil: ao invés de realizar determinadas operações diretamente (como no caso do Apple), ele efetua desvios e testes para checar se a máquina foi configurada de outro modo, de maneira a poder utilizar periféricos diferentes do usual. Podemos, por exemplo, ligar um MSX a uma impressora não compatível e configurá-lo de maneira a fazê-lo produzir, assim mesmo, os caracteres acentuados da língua portuguesa. Ou então podemos escrever na tela do vídeo com um conjunto de caracteres diferentes do residente. Quem já está familiarizado com os famosos "hooks" do MSX, sabe da extrema utilidade destes desvios. Eles têm um custo de tempo. O MSX, porém, é tão mais rápido que o Apple, que este tempo adicional é altamente compensado.

Há algum tempo, numa análise tendenciosa publicada no suplemento de Informática da Folha de São Paulo, o MSX foi apresentado como uma máquina muito mais lenta que o Apple.

Para fundamentar esta argumentação, o articulista mediu os tempos que o Apple e o MSX gastavam para achar certa quantidade de números primos. O algoritmo empregado não foi publicado, mas, segundo o articulista, o MSX empregava um tempo muito maior que o Apple. Para checar esta informação elaborei um curto programa em Basic residente (do MSX e do Apple) que identifica os números primos entre 2 e 2000 (veja listagem na figura 1), utilizando um algoritmo concebido por um especialista em computação há uns 2.200 anos atrás, um tal de Eratóstenes, que utilizava um hardware muito pouco empregado nos dias de hoje e que não dá para "clonar": o cérebro!

Rodando este programa em dois clones do Apple e num do MSX ligados a um drive, encontramos os seguintes valores:

Ap II TI (Unitron – Apple II+) – 66 segundos

TK 3000 IIe (Microdigital – Apple IIe) – 66 segundos

Expert 1.1 (Gradiente – MSX) – 61 segundos.

como podemos perceber, a diferença não é significativa. A igualdade de tempos de processamento ocorre apesar de todas as rotinas de desvio que o MSX faz e apesar do fato que a precisão de cálculo do MSX é muito maior que a do Apple. Enquanto o Apple realiza suas operações aritméticas em binário, o MSX as faz em BCD, recurso usado apenas por poucos outros micros, como o IBM-PC.

Se levarmos em conta que um algoritmo como o indicado funciona para números inteiros, podemos impor esta condição inicial digitando uma linha adicional:

100 DEFINT A-Z

Este recurso não existe no Apple (o Basic residente do MSX é incomparavelmente mais completo e poderoso que o do Apple).

Rodando esta nova versão do programa no MSX, encontramos:

Expert 1.1 (Gradiente – MSX) – 38 segundos, o que encerra definitivamente o assunto com relação à velocidade de processamento.

CUSTOS DO EQUIPAMENTO

A rigor, não tem muito sentido se comparar preços de duas máquinas de

Figura 1

```
100 DEFINT A-Z
110 DIM A(2001)
120  FOR I=2 TO 1999
130   IF A(I)=0 THEN GOSUB 160
140  NEXT I
150 END
160 PRINT I;
170 FOR N=2*I TO 2000 STEP I
180  IF N+1 > 2000 THEN RETURN
190   A(N)=1
200 NEXT N
210 RETURN
```

(com drive), a vantagem facilmente desaparece, pois o MSX, sem necessidade de placas adicionais, roda toda a gigantesca biblioteca de software desenvolvida em CP/M.

Se não quisermos ficar na simples pirataria de software desenvolvido lá fora e quisermos analisar a facilidade de produzir programas brasileiros, vemos que o MSX, por ser uma máquina muito veloz e versátil, permite uma agilidade de elaboração de novos programas incomparavelmente mais cômoda que o Apple.

Resumindo, o MSX é um equipamento que apareceu no mercado de microinformática do Brasil incomodando muita gente. O crescimento do sistema, com o surgimento de periféricos e software profissional, provocou tantas e estranhas reações que só podemos concluir que os incomodados tinham todas as razões do mundo para assim se sentir!

recursos tão distintos e de tecnologias tão defasadas no tempo. Apenas a título de curiosidade, propomos o exercício para o leitor, bastando consultar a figura 2, onde reproduzimos dois anúncios publicados no dia 19 de março de 1989 no jornal O Estado de São Paulo. Enquanto que um Hotbit, já interfaciado de fábrica para impressora, com Z-80 residente e 80 Kbytes de RAM custa NCz$ 369,00, um Apple TK 3000, com os mesmos recursos e apenas 64 Kbytes de RAM, custa 460,00 + 102,00 + 62,00 = NCz$ 624,00, ou seja, 69,1% a mais!

SOFTWARE DISPONÍVEL

Neste item, o Apple, por ser uma máquina mais antiga, aparentemente ganha de longe do MSX. Se analisarmos, porém, uma configuração profissional

TURBO PASCAL NOVAS APLICAÇÕES

NORBERTO TSOULESSKI

Depois de dois anos e meio do aparecimento dos microcomputadores MSX no Brasil, muitos são os usuários desta linha que possuem um acionador de discos, mesmo com os preços proibitivos dos periféricos. Com isso, é grande o número de usuários que possuem em suas casas compiladores das mais diversas linguagens, tais como Basic, Cobol, Pascal, C, etc.

Apesar disto, ainda é ínfimo o número de usuários que conhece profundamente uma destas linguagens e consegue elaborar programas de nível profissional usando o seu MSX e um compilador qualquer.

A linguagem estruturada mais conhecida dos usuários de MSX é, sem dúvida nenhuma, o Turbo Pascal. Mas, mesmo assim, não conheço nenhum usuário que a utilize para construir sistemas, criar um programinha utilitário para o DOS ou mesmo uma rotina para cálculos financeiros.

Sem dúvida, podemos contar nos dedos os programadores de MSX que utilizam o Turbo Pascal no seu dia-a-dia. Isto se deve à falta de informação sobre o Turbo Pascal, principalmente no que se refere ao MSX, pois existem muitos livros para PC e Apple, mas nenhum para a linha MSX. O resultado disto é que as pessoas interessadas se vêem impedidas de elaborar um programa mais complexo em Pascal, pois os exemplos de programas mais complexos que existem para os Apples e PC's não podem, na maioria das vezes, ser aplicados aos MSX.

A dificuldade existente em adaptar-se um programa escrito em um PC deve-se à quantidade de memória ocupada e também ao fato de que muitos livros apresentam programas que se utilizam de linguagem de máquina, lista com códigos de caracteres diferentes dos códigos do padrão MSX, comandos que só existem na versão 4.0 do Turbo Pascal e não possuem equivalentes na versão 3.0 usada pelo MSX, etc.

A falta de literatura sobre Turbo Pascal no MSX impede que os usuários se utilizem dos comandos mais interessantes que não existiam na linguagem Pascal e foram incluídos na versão Turbo, tais como: INLINE, EXTERNAL, BDOS, BIOS e ABSOLUTE.

Sendo assim, tomei a iniciativa de passar aos demais usuários de MSX o resultado de minhas experiências com os comandos de baixo nível do Turbo Pascal, mais precisamente com o comando INLINE.

O comando INLINE tem por função executar instruções em Assembler dentro de um programa Pascal. Além disso, ele permite que coloquemos dentro dele nomes de variáveis, o que torna a passagem de parâmetros extremamente fácil.

Utilizando o INLINE e conhecendo muito bem a estrutura do interpretador Basic, a linguagem Pascal e o Assembly, elaborei o programa que, a seguir, passarei a analisar.

Trata-se de um programa utilitário escrito em Turbo Pascal para o MSX e que utiliza rotinas em código de máquina do Z80 para criar novos e interessantes comandos para o Pascal.

O programa, na verdade um include file (arquivo que é incluído somente na hora da compilação), utiliza-se das rotinas em código de máquina para acessar o interpretador Basic que fica num outro slot do MSX e chamar as rotinas da ROM que o programa precisa para "simular" os mais poderosos comandos do MSX-BASIC.

Com este programa utilitário, qualquer pessoa pode confeccionar gráficos em alta resolução, executar uma música bem ao estilo do MSX ou até mesmo uma animação com sprites verdadeiros!

Você poderá acrescentar desenhos e sons aos programas que já possui ou até mesmo criar programas inteiramente novos utilizando instruções que antes só eram privilégio dos programadores de MSX-BASIC ou de Pascal de outras linhas de micros.

Apresentarei, agora, o programa tipo include que chamei de EXPANDE.PAS

PROCEDURE TEXTO(LIN,COL :INTEGER;TEXTO) :LITERAL); – coloca um texto na tela gráfica a partir da posição dada pelas variáveis LIN e COL.

FUNCTION VPEEK (X :INTEGER) :INTEGER; – retorna o conteúdo do endereço X da VRAM.

PROCEDURE SPRITE(N :INTEGER;TABELA:STRING[8]); – cria na VRAM um sprite 8x8 de acordo com o conteúdo da tabela.

PROCEDURE PUT_SPRITE(X,Y, N,C:INTEGER); – coloca o sprite dado por N e cor C na posição X,Y.

PROCEDURE SPRITE_AMPLIADO; – liga sprite 8x8 ampliado.

PROCEDURE SPRITE_NORMAL; – desliga sprite 8x8 ampliado.

PROCEDURE PLAY(MUSICAL:LITERAL); – toca seqüências de notas musicais com tempo, oitava, tom e volume programável. Possui sintaxe idêntica ao comando do MSX-BASIC. Usa um único canal de som.

PROCEDURE PSG(REG,CONTEÚDO:INTEGER); – grava o byte dado por conteúdo no registro REG do PSG (Gerador de sons programável). Sua utilização é a mesma do comando SOUND do MSX-BASIC.

Além destas, existem outras PROCEDURES auxiliares, que são usadas apenas para executar tarefas das mais diversas para as PROCEDURES principais que acabamos de ver.

São as seguintes as PROCEDURES auxiliares:

PROCEDURE WRITEMEM(ENDEREÇO:INTEGER;TEXTO:LITERAL); – serve para escrever o TEXTO na RAM a partir de ENDEREÇO.

PROCEDURE PROGRAME(DADOS:LITERAL); – coloca o programa em linguagem de máquina contido na string DADOSX na RAM, juntamente com os procedimentos para ativar e desativar o interpretador BASIC que permanece em um outro slot. A seguir, executa o programa.

PROCEDURE ERRO(NUM: BYTE); – interrompe o programa e imprime uma mensagem de erro, caso ocorra algum erro previsto durante o processamento.

PROCEDURE LDIRMV(CONT :INTEGER); – transfere 1024 bytes da VRAM para a RAM, a partir do endereço CONT da VRAM, usada pela procedure SAVE_TELA.

PROCEDURE LDIRVM(CONT :INTEGER); – transfere 1024 bytes da RAM para a VRAM, a partir do endereço CONT da VRAM. Usada pela (listagem 1) e também exemplos de utilização do mesmo (listagens 2, 3 e 4).

Trata-se, a bem da verdade, de uma coletânea de PROCEDURES e FUNCTIONS que não têm, obrigatoriamente, de ser usadas juntas, podendo fazer parte de uma biblioteca e, posteriormente, serem pesquisadas e utilizadas apenas aquelas necessárias para o programa que estiver sendo criado. Apesar disso, é aconselhável que se digite o programa da maneira que se apresenta e o inclua nos seus programas sempre que se precise, mesmo porque não é um programa tão longo.

São as seguintes as PROCEDURES e FUNCTIONS existentes:

PROCEDURE BEEP; – emite um sinal sonoro ("beep")

PROCEDURE COLOR(C1,C2,C3 :INTEGER); modifica as cores da frente, do fundo e da borda da tela.

PROCEDURE SCREEN(NUM :INTEGER); – seleciona um dos três modos da tela: MODO 0 (texto 40 colunas); MODO 1 (texto 32 colunas); MODO 2 (gráfico 256 x 192).

PROCEDURE PSET(X,Y,C :INTEGER); – acende o pixel dado pela coordenada X,Y dando ao mesmo a cor de C.

PROCEDURE LINE(X1,Y1,X2,Y2 :INTEGER); – desenha uma linha começando na posição dada por X1,Y1 e terminando na posição dada por X2,Y2".

PROCEDURE CIRCLE(X,Y,R,C :INTEGER); – desenha um círculo de raio R, centro em X,Y e cor C.

PROCEDURE PAINT(X,Y,C); pinta uma área da tela começando na coordenada dada por X,Y com a cor C, só terminando após encontrar-se cercada por pontos da mesma cor de C.

PROCEDURE DRAW(DESENHO :LITERAL); – desenha figuras na tela gráfica. A variável do tipo string DESENHO deve conter os subcomandos apropriados para o tipo de figura que se quer traçar. Sua sintaxe é igual ao comando DRAW do MSX-BASIC.

PROCEDURE SAVE–TELA(NOME :STRING[12]); – salva no disco todo o conteúdo da VRAM (16384 bytes), incluindo os desenhos e os sprites.

PROCEDURE LOAD_TELA (NOME:STRING[12]); – carrega no disco o conteúdo da VRAM salvo pela instrução SAVE_TELA.

PROCEDURE IMPRIME_TELA; – imprime na impressora a tela gráfica que está sendo mostrada no vídeo. A impressora deve seguir o padrão EPSON. O desenho será impresso na horizontal.

PROCEDURE LOAD_TELA.

PROCEDURE FILVRM(X:INTEGER); – preenche toda a tabela de cores da SCREEN 2 do MSX com o valor dado por X. Usado pela PROCEDURE IMPRIME_TELA.

PROCEDURE PRT(X:INTEGER); – envia o byte X para a impressora, servindo para contornar o problema de WRITE (LST, 'TEXTO'), só funcionando em SCREEN 0.

Os programas em TURBO PASCAL que forem utilizar estes novos gráficos e sonoros deverão ter a seguinte linha após a declaração das variáveis ou da palavra PROGRAM (nome);:

9=(*$I EXPANDE.PAS *)

Verifique os exemplos (listagens 2, 3 e 4) para tirar qualquer dúvida.

O primeiro exemplo é um programa de apresentação que demonstra toda a versatilidade do Turbo Pascal expandido, usando todos os recursos para desenhar linhas e círculos, usando as cores do MSX e também o comando para escrever na tela gráfica. No final, grava a tela no disco.

O segundo programa carrega a tela gravada pelo programa anterior e deixa-a à vista, enquanto executa uma linda melodia.

O terceiro exemplo mostra o comando PSG na geração de efeitos sonoros (um trem em movimento e uma explosão).

Para finalizar, algumas observações importantes:

- Existem 7 erros que foram previstos na elaboração do EXPANDE. É muito difícil ocorrer algum erro fora destes para o qual o programa não possui rotina de aviso.

– Somente os comandos PLAY e DRAW, por serem muito complexos, não possuem previsão de erro. Se ocorrer alguma falha quando na chamada destes comandos pelo programa que utilize o utilitário EXPANDE, ou seja, um erro de sintaxe, então pode dizer adeus ao seu programa.

– Pelo que disse na observação acima, tome cuidado na digitação dos programas que usarem PLAY e DRAW. Grave-os sempre antes de compilar.

– Neste momento, estou elaborando uma versão 2.0, mais sofisticada e que não possui o inconveniente de parar o micro quando ocorrer um erro de sintaxe em PLAY ou DRAW.

■

```
(* PROGRAMA UTILITARIO PARA TURBO  PASCAL  *)

(* ESTE PROGRAMA ACRESCENTA UMA SERIE   DE *)
(* NOVOS COMANDOS AO TURBO PASCAL QUE PER- *)
(* MITEM O DESENHO DE LINHAS  E  CIRCULOS, *)
(* EXECUCAO DE MUSICAS E  MANIPULACAO   DE *)
(* SPRITES ,ENTRE OUTRAS COISAS.           *)


(*  CREATED BY NORBERTO TSOULEFSKI  JAN/89 *)



(*$I-*)
(*$U-*) (* DIRETIVAS OPCIONAIS *)
(*$C-*)

TYPE LITERAL= STRING [80];
     AREA_BUFFER = ARR9Y [1..1024] OF BYTE;
     STR8 = STRING[8];
     STR12 = STRING[12];


VAR DADOS1,DADOS2,DADOSX : LITERAL;
    XA,XB,YA,YB,CS,RS : STRING [3];

PROCEDURE WRITEMEM(ENDERECO:INTEGER;TEXTO:LITERAL);
VAR INDICE,COMPRIMENTO:INTEGER;
BEGIN
  COMPRIMENTO:= LENGTH (TEXTO);
  FOR INDICE:=1 TO COMPRIMENTO DO
  BEGIN
    MEM [ENDERECO]:= ORD(TEXTO[INDICE]);
    ENDERECO:= ENDERECO+1;END;
    MEM [ENDERECO]:= 0
END;


PROCEDURE PROGRAME (DADOSX:LITERAL);
VAR SLOT: INTEGER;
BEGIN
IF PORT[168] = $FF  THEN SLOT:=$F0
   ELSE SLOT:= $A0;

  DADOS1:= #$DB+#$A8+#$F5+#$3E+CHR(SLOT)+#$D3+#$A8;
  DADOS2:= #$F1+#$D3+#$A8+#$C9;
  DADOSX:= DADOS1+DADOSX+DADOS2;

  WRITEMEM ($B100,DADOSX);
  INLINE ($C3/$00/$B1);
END;

PROCEDURE BEEP;
BEGIN
  DADOSX:=#$CD+#$C0+#00;
  PROGRAME(DADOSX);
END;

PROCEDURE ERRO(NUM:BYTE);
BEGIN
   DADOSX:=#$CD+#$6C+#00;
   PROGRAME(DADOSX);
   WRITE(CHR(7));

   CASE NUM OF
     1:WRITE('ERRO EM COLOR');
     2:WRITE('SCREEN INVALIDA');
     3:WRITE('ERRO EM CIRCLE');
     4:WRITE('ERRO EM PAINT');
     5:WRITE('OVERFLOW EM PSET');
     6:WRITE('ERRO EM LINE');
     7:WRITE('MODO GRAFICO REQUERIDO');
  END;
  HALT;
END;


PROCEDURE PSET( X,Y,C:INTEGER);
VAR DADOS:LITERAL;

BEGIN
  IF (X<0) OR (X>256) OR
     (Y<0) OR (Y>256) OR
     (C<0) OR (C>15) THEN ERRO(5);

  IF MEM[$FCAF] <> 2 THEN ERRO(7);


  STR(X,XA);
  STR(Y,YA);
  STR(C,CS);
  DADOS:='('+XA+','+YA+'),'+CS;
  WRITEMEM($B000,DADOS);
  DADOSX:=#$21+#00+#$B0+#$CD+#$EA+#$57;
  PROGRAME(DADOSX);
END;


PROCEDURE LINE(X1,Y1,X2,Y2,C:INTEGER);
VAR DADOS:LITERAL;
BEGIN
  IF (X1<0) OR (X1>256) OR (Y1<0) OR (Y1>192) OR
     (X2<0) OR (X2>256) OR (Y2<0) OR (Y2>192)
     OR (C<0) OR (C>15) THEN ERRO(6);

  IF MEM[$FCAF] <> 2 THEN ERRO(7);


  STR(X1,XA);
  STR(Y1,YA);
  STR(X2,XB);
  STR(Y2,YB);
  STR(C,CS);
  DADOS:='('+XA+','+YA+')'+CHR($F2)+'('+XB+','+YB+'),'+CS;
  WRITEMEM($B000,DADOS);
  DADOSX:=#$21+#00+#$B0+#$CD+#$0E+#$4B;
  PROGRAME(DADOSX);
END;
PROCEDURE PLAY (MUSICA:LITERAL);
BEGIN
  MEM[$B000]:=34;
  WRITEMEM($B001,MUSICA+CHR(34));
  DADOSX:=#$21+#00+#$B0+#$CD+#$E5+#$73;

  PROGRAME(DADOSX);
END;

PROCEDURE SCREEN (NUM:INTEGER);
BEGIN
  CASE NUM OF
    0 : DADOSX:=#$CD+#$6C+#00;
    1 : DADOSX:=#$CD+#$6F+#00;
    2 : DADOSX:=#$CD+#$72+#00;
  ELSE
    ERRO(2);
  END;
  PROGRAME(DADOSX);
END;


PROCEDURE COLOR(C1,C2,C3:INTEGER);
BEGIN
  IF (C1<0) OR (C1>15) OR
     (C2<0) OR (C2>15) OR
     (C3<0) OR (C3>15) THEN ERRO(1);


  MEM[$F3E9]:= C1;
  MEM[$F3EA]:= C2;
  MEM[$F3EB]:= C3;
  DADOSX:= #$CD+#$62+#00;
  PROGRAME (DADOSX);
  END;


PROCEDURE DRAW (DESENHO:LITERAL);
BEGIN
  IF MEM[$FCAF] <> 2 THEN ERRO(7);
  MEM[$B000]:=34;
  WRITEMEM($B001,DESENHO+CHR(34));
  DADOSX:= #$21+#00+#$B0+#$CD+#$6E+#$5D;
  PROGRAME(DADOSX);

END;


PROCEDURE CIRCLE (X,Y,R,C:INTEGER);
VAR
DADOS:LITERAL;

BEGIN
  IF (X<0) OR (X>256) OR (Y<0) OR (Y>192) OR
     (R<0) OR (R>200) OR (C<0) OR (C>15)
     THEN ERRO(3);

  IF MEM[$FCAF] <> 2 THEN ERRO(7);

  STR(X,XA);
  STR(Y,YA);
  STR(R,RS);
```

```
   STR(C,CS);
   DADOS:='('+XA+','+YA+')'+','+RS+','+CS;
   WRITEMEM ($B000,DADOS);
   DADOSX:= #$21+#00+#$B0+#$CD+#$11+#$5B;
   PROGRAME (DADOSX);

 END;


 PROCEDURE PAINT (X,Y,C:INTEGER);
 VAR
 DADOS:LITERAL;
 BEGIN
   IF (X<0) OR (X>256) OR
      (Y<0) OR (Y>192) OR
      (C<0) OR (C>15) THEN ERRO(4);


   IF MEM[$FCAF] <> 2 THEN ERRO(7);

  STR(X,XA);
  STR(Y,YA);
  STR(C,CS);
  DADOS:='('+XA+','+YA+')'+','+CS;
  WRITEMEM ($B000,DADOS);
  DADOSX:= #$21+#00+#$B0+#$CD+
          #$C5+#$59;
  PROGRAME (DADOSX);
END;


PROCEDURE TEXTO(LIN,COL:INTEGER;TEXTO:LITERAL);
VAR C:BYTE ABSOLUTE $F3EA;
INDICE,COMPRIMENTO: INTEGER;
LETRA : CHAR;
BEGIN
  IF MEM[$FCAF] <> 2 THEN ERRO(7);

  COMPRIMENTO:=LENGTH(TEXTO);
    FOR INDICE:=1 TO COMPRIMENTO DO
    BEGIN
        LETRA:=TEXTO[INDICE];
        PSET(COL*6,LIN*8,C);
        DADOSX:=#$3E+LETRA+#$CD+#$8D+#00;
        PROGRAME(DADOSX);
        COL:=SUCC(COL);
        IF COL>40 THEN BEGIN
            COL:=0;
            LIN:=SUCC(LIN);
        END;
    END;
END;

PROCEDURE SPRITE(N:INTEGER;TABELA:STR8);
BEGIN
  WRITEMEM($B000,TABELA);
  DADOSX:=#$3E+CHR(N)+#$CD+#$84+#00+#$EB+#$21+
          #00+#$B0+#01+#08+#00+#$CD+#$5C+#00;
  PROGRAME(DADOSX);
END;

PROCEDURE PUT_SPRITE (X,Y,N,C:INTEGER);
BEGIN
  DADOSX:=#$3E+CHR(N)+#$CD+#$87+#00+#$3E+CHR(Y)+#$CD+
          #$4D+#00+#$23+#$3E+CHR(X)+#$CD+#$4D+#00+
          #$23+#$3E+CHR(N)+#$CD+#$4D+#00+#$23+#$3E+
          CHR(C)+#$CD+#$4D+#00;
  PROGRAME (DADOSX);
END;


PROCEDURE SPRITE_AMPLIADO;
BEGIN
  DADOSX:=#$3A+#$E0+#$F3+#$CB+#$C7+#$47+
          #$0E+#01+#$CD+#$47+#00:
  PROGRAME(DADOSX);
END;


PROCEDURE SPRITE_NORMAL·
BEGIN
  DADOSX:=#$3A+#$E0+#$F3+#$CB+#$87+#$47+
          #$0E+#01+#$CD+#$47+#00;
  PROGRAME(DADOSX);
END;


PROCEDURE PSG(REG,CONTEUDO :INTEGER);
BEGIN
  PORT[160]:=REG;
  PORT[161]:=CONTEUDO;
END;

PROCEDURE LDIRMV (CONT:INTEGER);
BEGIN
  DADOSX:= #$11+#00+#$B2+#$21+CHR(LO(CONT))+CHR(HI(CONT))
        +#01+#00+#04+#$CD+#$59+#00;
  PROGRAME (DADOSX);
END;


PROCEDURE LDIRVM (CONT:INTEGER);
BEGIN
  DADOSX:= #$21+#00+#$B2+#$11+CHR(LO(CONT))+CHR(HI(CONT))
         + #01+#00+#04+#$CD+#$5C+#$00;
  PROGRAME(DADOSX);
END;


PROCEDURE SAVE_TELA (NOME:STR12);
VAR BUFFER : AREA_BUFFER ABSOLUTE $B200;
   TELA : FILE OF AREA_BUFFER;
CONT :INTEGER;

BEGIN
 CONT:= 0 ;
 ASSIGN (TELA,NOME);
  REWRITE(TELA);
 WHILE CONT < 16384 DO
 BEGIN
   LDIRMV(CONT);
   WRITE(TELA,BUFFER);
   CONT:= CONT +1024;
 END;
 CLOSE (TELA);
END;


PROCEDURE LOAD_TELA (NOME:STR12);
VAR
 BUFFER:AREA_BUFFER ABSOLUTE $B200;
 TELA:FILE OF AREA_BUFFER;
 CONT : INTEGER;
BEGIN
  CONT:=0;
  ASSIGN (TELA,NOME);
  RESET(TELA);
  WHILE CONT < 16384 DO
  BEGIN
    READ (TELA,BUFFER);
    LDIRVM(CONT);
    CONT:=CONT + 1024;
  END;
  CLOSE (TELA);
END;


FUNCTION VPEEK(X:INTEGER):INTEGER;
BEGIN
  DADOSX:=#$21+CHR(LO(X))+CHR(HI(X))+#$CD+#$4A+#$00
         +#$32+#$00+#$F4;
  PROGRAME (DADOSX);
  VPEEK:=MEM[$F400];
END;


 PROCEDURE FILVRM (X:INTEGER);
 BEGIN
   DADOSX:=#$3E+CHR(X)+#$21+#$00+#$20
          +#$01+#$00+#$18+#$CD+#$56+#$00;
   PROGRAME (DADOSX);
 END;
 PROCEDURE PRT(X:INTEGER);
 BEGIN
   DADOSX:=#$3E+CHR(X)+#$CD+#$A5+#00;
   PROGRAME (DADOSX);
 END;

PROCEDURE IMPRIME_TELA ;
VAR
COL,LIN,X,U,V:INTEGER;

BEGIN
  FILVRM(31);
  PRT(27);PRT(65);PRT(8);

  FOR COL:= 0 TO 31 DO
  BEGIN
    PRT(27);PRT(75);PRT(192);PRT(0);
    FOR LIN:= 23 DOWNTO  0 DO
```

```
            BEGIN
              FOR X:= 7 DOWNTO 0 DO
              BEGIN
                U:= VPEEK((COL*8+256*LIN)+X);
                V:= VPEEK((COL*8+256*LIN)+X+8192);
                IF V MOD 16 = 1 THEN U:=255;
                PRT(U);
              END;
            END;

              PRT(10);

          END;
          PRT(27);PRT(65);PRT(13);
          PRT(27);PRT(81);PRT(39);
          END;
```

```
PROGRAM MUSICA;
(*$I EXPANDE.PAS *)

  BEGIN
    COLOR(1,15,4);

    SCREEN(2);

    SPRITE_AMPLIADO;

    LOAD_TELA ('ABERTURA');

    PLAY('V15T190L4O4GABO5DCCEDDGF#GDO4BGABO5CDEDCO4BABG');


PLAY('V15T190L4O4F#GADF#AO5CO4BABGABO5DCCEDDGF#GDO4BGABEO5DCO4BAG
DGF#');


PLAY('V15T190L4O4GBO5DGDO4BGBO5DFDO4BGBO5DECO4AF#AO5CDO4BGEGBO5CO
4AF#DF#AO5CO4B');


PLAY('V15T190L4O4BGABO5DCCEDDGF#GDO4BGABO5CDEDCO4BABGF#GADF#AO5CO
4BA');


PLAY('V15T190L4O4BGABO5DCCEDDGF#GDO4BGABEO5DCO4BAGDGF#O4L2G.');

     SCREEN(0);

  END.
```

```
PROGRAM GRAFICO:
VAR I,X,Y,C1,C2,C3,C4 :INTEGER;
    TABELA :STRING[8];

(*$I EXPANDE.PAS *)

PROCEDURE MUSICA;
BEGIN
PLAY('T255L8ABDCDEEA');
END;

BEGIN
COLOR (8,15,4);
SCREEN(2);
X:=0;
TEXTO(1,4,'GRAFICOS');
TEXTO(3,7,'NO');
TEXTO(5,3,'TURBO PASCAL');
REPEAT
   Y:=TRUNC(80-((X*80)/125));
   C1:=2+RANDOM(13);
   C2:=2+RANDOM(13);
   C3:=2+RANDOM(13);
   C4:=2+RANDOM(13);
   LINE(X,80,125,Y,C1);
   LINE(125,Y,250-X,80,C2);
   LINE(X,80,125,160-Y,C3);
   LINE(125,160-Y,250-X,80,C4);
    X:=X+10;
UNTIL X > 125;
TABELA:=#102+#102+#102+#102+#219+#219+#219+#219;
SPRITE(1,TABELA);
TABELA:=#127+#255+#192+#254+#127+#3+#255+#254;
SPRITE(2,TABELA);
TABELA:=#193+#227+#119+#62+#28+#62+#119+#99;
SPRITE(3,TABELA);
SPRITE_AMPLIADO;

FOR I:=1 TO 170 DO
BEGIN
  PUT_SPRITE(102,I,1,1);
  PUT_SPRITE(102+16,I,2,1);
  PUT_SPRITE(102+32,I,3,1);

END;
DRAW('C11BM13,124D14R70U14L70');
DRAW('C11BM165,124D14R70U14L70');
PAINT(180,130,11);
PAINT(40,130,11);
COLOR(1,15,4);
TEXTO(16,4,'NORBERTO');
TEXTO(16,28,'TSOULEFSKI');
SAVE_TELA('ABERTURA');
REPEAT MUSICA UNTIL KEYPRESSED;
SCREEN(0);
END.
```

```
PROGRAM SOUND;
(*$I EXPANDE.PAS *)
VAR
  REG : INTEGER;
CONST
  BOMBA : ARRAY [0..13] OF INTEGER=
  (0,0,0,0,0,0,21,247,16,0,0,100,60,0);
  TREM  : ARRAY [0..13] OF INTEGER=
  (0,0,0,0,255,15,8,3,16,7,16,90,4,8);
BEGIN
REG:=0;
  WRITELN ('TREM');
  REPEAT
    PSG(REG,TREM[REG]);
    REG:= REG + 1;
    WRITE (PORT[162],' ');
 UNTIL REG > 13;
 DELAY (4000);
REG:= 0;
  WRITELN;
  WRITELN ('BOMBA');
  REPEAT
    PSG(REG,BOMBA[REG]);
    REG:=REG+1;
    WRITE (PORT[162],' ');
  UNTIL REG > 13;
END.
```

D R A G O N   N I N J A

Simplesmente o melhor até o momento em lutas marciais !

Apenas em disco por NCz$ 9,00

SPECIAL GAME PACK 13

PAC-LAND, BESTIAL WARRIOR, e THE PINK PANTHER reunidos num mesmo disco - NCz$ 15,00.

NOVIDADES PARA MSX2 64 KB

LEATHER SKIRTS, TRIDIMENSIONAL CRAFTON e AGATH ARKANOID
CADA UM EM DISCO NCz$ 9,00.

CAPAS PARA COMPUTADORES

Proteja o seu equipamento da poeira e da umidade:

EXPERT (CPU+TECLADO) .Cz$ 18,00
HOTBIT ou TALENT....Cz$ 12,00
CAPA PARA DRIVE.....Cz$ 8,00
CAPA PARA LADY/MTA..Cz$ 12,00

NEMESIS INFORMATICA LTDA.

Envie VALE POSTAL ou CHEQUE NOMINAL a NEMESIS INFORMATICA Caixa postal 4.583 Cep 20.001 Rio de Janeiro - RJ ou venha pessoalmente na: Rua Sete de Setembro 92/1910 CENTRO - RJ.

T H E   M U N S T E R S

A NEMESIS traz até você as últimas novidades em jogos e aplicativos para a linha MSX:

Da TV: A Família Monstro!

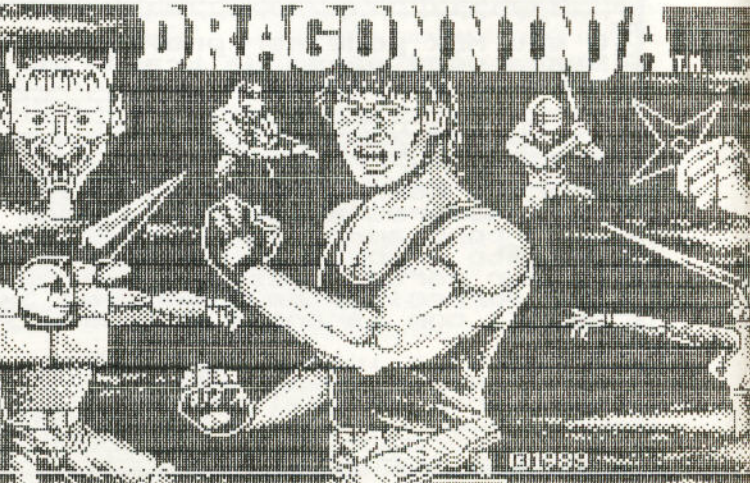
Apenas em disco por NCz$ 8,00

4 X 4 - OFF ROAD RACING

Monte sua PICK-UP e enfrente um RALLY no deserto !!!

Apenas em disco por NCz$ 8,00

AGUARDE OS LANÇAMENTOS: DOUBLE DRAGON E AFTERBURNER

NOVIDADES P/ MSX PAGE MAKER

MSX PAGE MAKER FONTS 3 e PAGE MAKER FONTS 4. Mais uma variedade inédita de:

LETRAS, LETRAS, *LETRAS*, ETC.

MSX PAGE MAKER CARTOONS 2 e MSX PAGE MAKER TITLES nº 1 com uma infinidade de figuras para compor suas artes:

CADA UM EM DISCO - Cz$ 12,40.

MSX PAGE MAKER

À NEMESIS INFORMATICA está lançando a nova versão do melhor aplicativo já desenvolvido para os micro-computadores MSX :

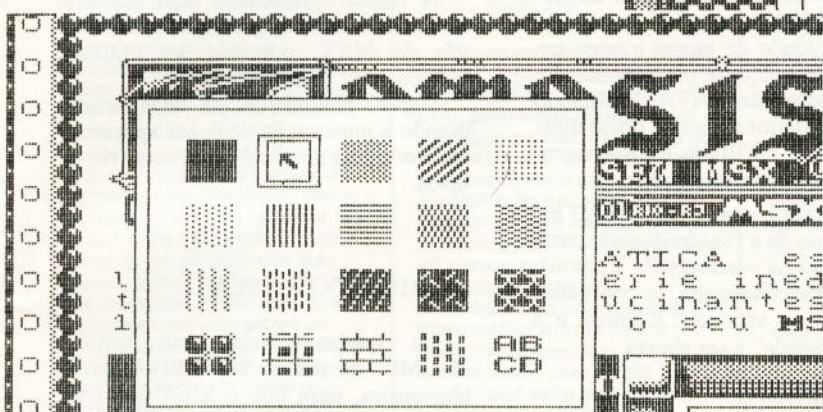
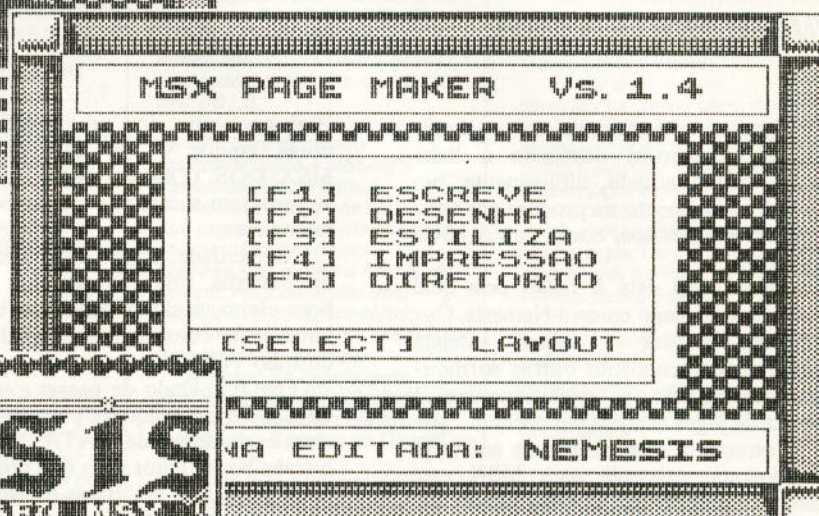
MSX PAGE MAKER 1.4

Mais que um processador, ele reúne textos e gráficos para o desenvolvimento de páginas super especiais para as mais diversas utilidades. Dê asas a sua imaginação !!

MSX PAGE MAKER Vs. 1.4

[F1] ESCREVE
[F2] DESENHA
[F3] ESTILIZA
[F4] IMPRESSAO
[F5] DIRETORIO

[SELECT] LAYOUT

Os módulos principais do programa são: o REDATOR, responsável por toda a parte escrita; o DESENHISTA, com recursos gráficos ilimitados...

Temos ainda o estilista, que prepara a página a ser editada com diferentes bordas, títulos, cabeçalhos, e uma infinidade de acessórios !!!

O MSX PAGE MAKER 1.4 é ainda compatível com o GRAPHOS III e seus:

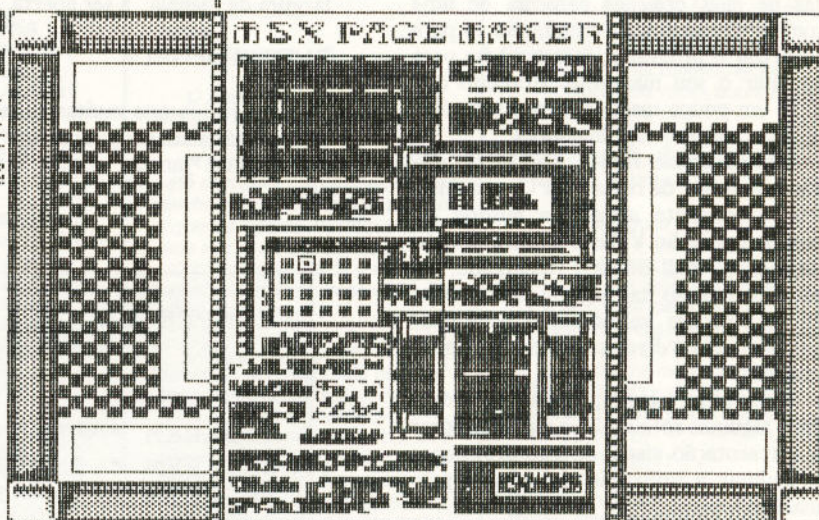
SHAPES

Figuras decorativas que podem ser colocadas facilmente em seus desenhos;

LETRAS

ALFABETOS, ALFABETOS, ALFABETOS, e diversos tipos de letras disponíveis, inclusive COMPACTADA e INVERTIDA, ou seja, centenas de combinações !

TELAS

Compatibilidade a nível de telas com todos os editores gráficos existentes no mercado para MSX, pois possui um inédito conversor interno para converter todos os padrões existentes.

MSX PAGE MAKER 1.4

NEMESIS INFORMATICA
CAIXA POSTAL 4.563
CEP. 20.004 - RJ
RIO DE JANEIRO

UM SOFTWARE MSX
100% NACIONAL

PROTEJA-SE CONTRA OS PIRATAS !

PROGRAMAÇÃO PROFISSIONAL NO MSX

MARCELO VALLE FRANCO

Como proprietário de um estabelecimento comercial destinado à linha MSX, tenho notado, ultimamente, um grande crescimento na procura por aplicativos e utilitários, como nunca havia ocorrido.

Talvez seja esta a razão pela qual empresas do ramo como a Nemesis, Cybertron, Practica e Paulisoft estejam progredindo, enquanto outras softhouses que dedicam-se exclusivamente à venda de jogos estejam em declínio.

Entretanto, a linha atual de software profissional existente para MSX está ainda carente, o que nos faz perder alguns clientes para os IBM-PC.

Por outro lado, está aberto um grande filão para os programadores que queiram produzir e comercializar tais produtos. Estes programadores estão realmente aparecendo. No entanto, muitas de suas criações carecem de uma melhor apresentação visual, não só para valorizar o programa, mas também para facilitar o seu manuseio. Acredito que falte um pouco mais de informação técnica para estes programadores.

Entusiasmado fiquei quando, no número anterior da revista CPU, encontrei um interessante artigo da autoria do meu amigo Julio Velloso sobre a criação de menus "pull-down". Realmente, tenho encontrado nas páginas desta revista informações valiosíssimas para programadores de diversos níveis de conhecimento.

Aproveito, então, para colaborar com algumas rotinas úteis para melhorar a apresentação visual de seus programas e facilitar a utilização do mesmo pelo usuário.

JANELAS EM LINGUAGEM DE MÁQUINA

A primeira rotina apresentada é para a criação de "janelas", que estão muito em moda atualmente.

O sistema de "janelas" que descrevo deve ser utilizado na SCREEN 0. Ele é o mesmo sistema utilizado por Eduardo Barbosa – um dos programadores de nossa equipe – em seus programas MSX-DOS TOOLS (BR) e Hello!, que se tornaram sucesso de vendas pela nossa empresa.

Na verdade, trata-se de uma rotina de extrema simplicidade, mas de um bom efeito visual, podendo ser utilizada juntamente com os menus "pull-down" de Júlio Velloso.

Tive o cuidado de passar a parte em linguagem de máquina para montagem através de linhas de DATA, facilitando o trabalho do leitor para que possa digitar o programa em Basic e dispensar o uso de um montador Assembler.

As linhas de comentários (REM) indicam claramente a função de cada parte do programa. As variáveis X, Y, A e L apontam, respectivamente, as coordenadas horizontais, verticais, a altura e a largura da "janela" a ser aberta.

BUFFER DE TELA

Uma rotina que muitas vezes torna-se necessária em programas que utilizem a técnica de "janelas" é a de "buffer" de tela.

Esta rotina é a responsável pela restauração da tela que foi parcialmente ou totalmente sobre-escrita pela "janela".

A rotina trabalha armazenando anteriormente a abertura da "janela", a tela em determinada posição de memória para representá-la em seguida em determinado momento.

A rotina apresentada aqui funciona adequadamente em qualquer formato de tela do MSX, podendo ser utilizada também com telas gráficas.

Digite, juntamente, as duas rotinas usando a numeração de linhas apresenada. Rode o conjunto e observe os resultados.

IMPRIMINDO A TELA

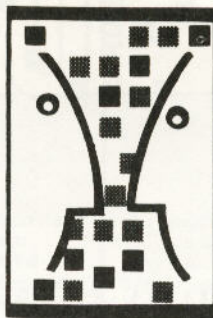
De quebra, incluo um simples "DUMP" de tela de SCREEN 0 para a impressora, para que o leitor possa ter no papel o resultado das rotinas anteriores.

Em breve, voltaremos a este espaço com mais dicas de programação para software profissional.

```
DUMPER DE TELA EM SCREEN 0

170 REM
460 REM ********************************
461 REM * IMPRIME A TELA EM SCREEN 0 *
462 REM ********************************
463 REM
464 FOR N = 0 TO 960
465 E = VPEEK ( BASE ( 0 ) + N )
466 LPRINT CHR$ ( E ) ;
467 IF L = 39 THEN LPRINT : L = 0 : NEXT
 ELSE 468
468 L = L + 1 : NEXT N
```

```
100 REM ********************************
110 REM *  ROTINA PARA ABERTURA DE     *
120 REM *       JANELAS DO TIPO        *
130 REM * MSX-DOS TOOLS (BR) e HELLO! *
140 REM *  NEMESIS INFORMATICA LTDA.  *
150 REM ********************************
160 REM
170 KEYOFF:SCREEN0:WIDTH40: REM INICIA
180 REM
190 REM ********************************
200 REM *   MONTA A ROTINA EM L.M.    *
210 REM ********************************
220 REM
230 FOR  N = &HDF00 TO &HE018
240 READ A$
250 J = VAL("&h"+A$)
260 T = T + J
270 POKE N,J
280 NEXT N
290 IF T<>31277 THEN PRINT"ERRO!": END
300 DEFUSR1 = &HDF00: REM END. ENTRADA
310 REM
320 REM ********************************
330 REM *  DETERMINA AS COORDENADAS   *
340 REM ********************************
350 REM
360 Y=3  :REM INFERIOR ESQUERDA
370 X=3  :REM SUPERIOR ESQUERDA
380 A=15 :REM INFERIOR DIREITA (ALTURA)
390 L=38 :REM SUPERIOR DIREITA (LARGURA)
400 REM
410 REM ********************************
420 REM *       ABRE A JANELA          *
430 REM ********************************
440 REM
450 POKE&HDF01,Y:POKE&HDF02,X:POKE&HDF04
,A:POKE&HDF05,L : U = USR1(0)
460 LOCATE0,18 : END : REM FINALIZA
470 REM
480 REM ********************************
490 REM *       LINHAS DE DATA         *
500 REM ********************************
510 REM
520 DATA 21,0,0,11,0,0,AF,32,E1,DF,32,E2
,DF,22,E3,DF,7C,82,CB,3F,3D,67,3C,3C,57,
7D,83,CB,3F,3D,6F,3C,3C,5F,CD,31,DF,CD,4
0,DF,CD,5D,DF,CD,4F,DF,C8,18,F1,3A,E4,DF
,BC,30,3,25,14,C9
530 DATA 3E,1,32,E1,DF,C9,3A,E3,DF,BD,30
,3,2D,1C,C9,3E,1,32,E2,DF,C9,3A,E1,DF,FE
,1,28,1,C9,3A,E2,DF,FE,1,C9,F3,E5,D5,E5,
CD,BE,DF,E1,7A,94,3D,3D,47,C5,3E,18,D3,9
8,3E,17,D3,98,10,FA,3E,19,D3,98,C1,6B,CD
,BE,DF,3E,1A,D3
540 DATA 98,3E,17,D3,98,10,FA,3E,1B,D3,9
8,D1,E1,E5,D5,2C,7B,95,47,7A,94,3D,3D,4F
,E5,CD,BE,DF,E1,3E,16,D3,98,C5,41,3E,20,
D3,98,10,FA,3E,16,D3,98,2C,C1,10,E7,FB,2
1,0,10,2B,7C,B5,20,FB,D1,E1,C9,F5,C5,D5,
25,2D,7C,F5,26
550 DATA 0,7D,5F,16,0,6,27,19,10,FD,F1,5
F,16,0,19,7D,D3,99,7C,F6,40,D3,99,D1,C1,
F1,C9,0,0,0,0,8E,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,21,0,80,11,
1,80,1,0,50,36,0,ED,B0,FD,2A,47,F3,DD,21
,22,40,C3,1C,0,46

170 REM
291 REM ******************************
292 REM *       BUFFER DE TELA       *
293 REM ******************************
294 REM
295 FOR N = &HEA60 TO &HEAB9 : READ A$
296 O = VAL("&H"+A$)
297 P = P + O
298 POKE N,O : NEXT N
299 IF P<>9347 THEN PRINT"ERRO!" : END
305 DEFUSR2 = &HEA60 : DEFUSR3 = &HEA8D
391 REM
392 REM ******************************
393 REM *      MEMORIZA A TELA       *
394 REM ******************************
395 REM
396 U = USR2(0)
460 REM
461 REM ******************************
462 REM *    RESTABELECE A TELA      *
463 REM ******************************
464 REM
465 U = USR3(0) : GOTO 450
560 REM
570 REM ******************************
580 REM *      LINHAS DE DATA        *
590 REM ******************************
600 REM
610 DATAF3,DB,A8,F5,FE,A0,28,C,FE,F0,28,
4,3E,54,18,6,3E,FC,18,2,3E,A8,D3,A8,21,0
,0,11,0,40,1,0,40,CD,59,0,F1,D3,A8,CD,24
,0,FB,C9,0,F3,DB,A8,F5,FE,A0,28,C,FE,F0,
28,4,3E,54,18,6,3E,FC,18,2,3E,A8,D3,A8,2
1,0
620 DATA40,11,0,0,1,0,40,CD,5C,0,F1,D3,A.
8,CD,24,0,F,C9,0
```

ENTREVISTA

• O que é a Nemesis?

A Nemesis é uma empresa que foi criada em abril de 1987 por um grupo que acompanhava o MSX desde o seu lançamento oficial no Japão, em 1983. O objetivo da Empresa é a representação, venda e desenvolvimento de software educativo, recreativo e profissional para microcomputadores.

• No Brasil a maioria do software existente é estrangeiro. Como é possível a comercialização destes programas perante a legislação atual? Qual a diferença entre o software estrangeiro e o nacional?

Desde que lançadas no Brasil, em 1985, as máquinas MSX (Expert da Gradiente e Hotbit da Sharp) tiveram suas campanhas publicitárias dirigidas aos jovens, tendo como "carros-chefes" os jogos deslumbrantes trazidos do exterior e adaptados pelos fabricantes aos MSX nacionais.

Em pouco tempo, os fabricantes pararam de importar e adaptar jogos e outras empresas se incumbiram desta tarefa, criando um mercado de suporte para os consumidores destas máquinas.

Existem três tipos de software para MSX no Brasil: o importado e adaptado, sem representantes legais no país, como é o caso da maioria dos jogos; o importado com representação legal, como é o caso do dBASE II e SUPERCALC 2; e os programas puramente nacionais registrados no Brasil por seus autores, como é o caso do nosso MSX PAGE MAKER, o FASTCOPY da Paulisoft, o EMULADOR ZX 81 da XSW e outros produtos de qualidade superior a de muitos estrangeiros.

Quanto à comercialização destes programas: os do primeiro grupo são considerados de domínio público em território brasileiro por não terem representantes legais e não estarem sujeitos às fiscalizações exercidas pelos seus distribuidores no Brasil; os do segundo e terceiro grupos somente podem ser comercializados pelas empresas responsáveis pelos mesmos ou por representantes legais das mesmas.

MARCELO VALLE FRANCO, 26 anos, formado pela Universidade Federeal do Rio de Janeiro, é o proprietário da NEMESIS INFORMÁTICA, a mais conhecida Softhouse MSX do país. Atualmente, a Nemesis tem seus esforços dirigidos no desenvolvimento de programas nacionais como o MSX PAGE MAKER. Além deste segmento, a Softhouse carioca é também conhecida pelo lançamento de jogos inéditos para MSX.

• Como a Nemesis investe no mercado de criação de software?

Como o mercado de software nacional para MSX ainda não alcançou as altas cifras, como aconteceu no mercado de software para microcomputadores da linha IBM-PC, os melhores programadores profissionais brasileiros ainda não recorreram a este filão. No entanto, temos certeza de que isso ocorrerá em breve.

Enquanto isso, temos estimulado um grande número de jovens interessados em programar para o MSX, dando o apoio suficiente para a conclusão de seus trabalhos e finalizando na compra dos direitos autorais pelos mesmos ou no direito de comercialização por meio de cópias numeradas.

• Investir em desenvolvimento de software tem sido um bom negócio para a Nemesis?

Sim, porque hoje e cada vez mais, as melhores empresas nacionais se destacam justamente pelo lançamento de software 100% nacional e, além disso, os mesmos trazem um faturamento muito maior que os importados ou adaptados.

• O que pode fazer um autor de software para poder comercializá-lo?

O primeiro passo é registrar o programa como de sua propriedade junto a algum órgão público, particular ou mesmo publicitário que o defenderá judicialmente pelos seus direitos. Depois, tentar comercializá-lo através de algum estabelecimento legalmente regularizado para o comércio deste tipo de produto. O comércio particular não legalizado pode acarretar infindáveis problemas. Antes de comercializar o seu produto, procure consultar um advogado, um contador ou uma softhouse de confiança.

• Como são pagos os direitos autorais a um criador de software?

Existem dois meios seguros para um programador comercializador um programa por intermédio de uma softhouse: a venda dos direitos, quando o programador vende os direitos de venda do seu

ENTREVISTA

programa para que a softhouse distribua-o livremente entre seus clientes; o segundo método é a cessão da venda de cópias, onde o programador recebe uma porcentagem pelo número de cópias vendidas pela softhouse.

No primeiro caso, o programador recebe imediatamente o valor integral pelo seu produto, e no segundo, que deve ser feito apenas com excessiva margem de segurança, o prazo de faturamento cresce.

• Quais são os prejuízos que a pirataria traz para o mercado de software nacional e para os usuários?

Numa época onde a própria existência e o desenvolvimento do MSX depende cada vez mais de produções brasileiras, eu diria que piratear software nacional é simplesmente destruir o sucesso da linha MSX no Brasil.

Eu fico curioso em saber porquê, com tanto joguinho de "domínio público" para ser vendido sem maiores conseqüências, a pirataria e as "pirato-houses" insistem em comercializar software registrado de origem nacional, o que pode dar, sem dificuldade, mais de 4 anos de detenção aos infratores.

• Como são feitos os lançamentos de jogos e como a Nemesis lança tantos títulos inéditos no Brasil?

Como não possuem representantes legais em nosso território, os jogos importados são adquiridos no exterior, adaptados e distribuídos livremente no Brasil por diversas empresas, entre elas, até mesmo, a Gradiente e a Sharp.

Neste segmento, destacam-se as empresas que conseguem títulos inéditos antes das demais. É puramente o caso de trabalhar melhor que os concorrentes.

• Atualmente, temos bons programas aplicativos e utilitários desenvolvidos no Brasil. Quando teremos um jogo do nível dos jogos da KONAMI, desenvolvidos por programadores brasileiros?

Como já foi dito, os aplicativos e utilitários nacionais rendem muito mais que os joguinhos. No momento, dificilmente, alguém ou alguma empresa se interessará em produzi-los ao nível que os mesmos são feitos no exterior.

• Grande parte dos usuários de MSX foi acostumado a comprar programas a preço de banana, ao contrário do que acontece no exterior, onde um programa pode custar tanto quanto o próprio computador, ou mesmo mais. Diante deste quadro, como se comportam as vendas de um KIT MSX PAGE MAKER ou de um SUPERCALC 2, que custam mais de NCz$ 100,00?

É necessário que se desenvolva no usuário, uma mentalidade mais madura quanto ao uso do microcomputador que adquiriu. Muitos dos meus clientes compraram um equipamento MSX completo simplesmente para usar o aplicativo dBASE II PLUS. Quem gastou mais de 2 mil cruzados em hardware não vai fazer cara feira ao pagar por volta de NCz$ 100,00 em uma cópia original deste software com documentação completa, garantia e suporte.

Se o usuário quiser simplesmente jogar, não vou censurá-lo quando correr atrás do melhor preço por um joguinho entre os distribuidores.

• Como definir o usuário de MSX hoje?

Ele já está com a mentalidade que falei anteriormente.

• O que ele procura na Nemesis?

Principalmente nossos produtos e aqueles que representamos de outras softhouses brasileiras. Já temos, hoje, uma proporção de 70% do faturamento em produto nacional.

• O mercado de MSX sempre foi um tanto tumultuado com problemas como lançamentos que não são efetuados pelos fabricantes de hardware, equipamentos que não funcionam adequadamente, incompatibilidade, assistências técnicas incapacitadas, etc. Que conselhos daria a um usuário de MSX para que tire vantagens deste mercado e possa utilizar seu micro com segurança?

A falta dos lançamentos prometidos pelos fabricantes foi intensamente contornada por empresas paralelas como a Microsol, DDX, Cibertron, Grafix, Elgin, Racidata, Telcom e Blump que, entre outras, fizeram um excelente trabalho e garantiram seu faturamento acreditando no sucesso do MSX no Brasil.

O conselho que eu daria aos usuários é de, simplesmente, procurar as melhores marcas e testar os produtos antes de levar para casa.

• A Nemesis tem instalado sistemas baseados no MSX em escritórios de Advocacia. Esta experiência tem dado certo?

Todos os meus clientes que tiveram seus estabelecimentos ou atividades informatizadas com o MSX estão satisfeitos. Não há como comparar a relação CUSTO/BENEFÍCIO do MSX com outras máquinas no caso da informatização de pequenas empresas.

• Na área profissional, do que o MSX é capaz e ainda não foi feito?

Eu sou simplesmente capaz de escrever um livro sobre este assunto. Será que o Pierluigi editaria?

• Quais os próximos lançamentos da Nemesis na área de software para MSX?

Temos planos de lançar um pacote integrado de Banco de dados, Planilha de cálculos e Processador de textos, uma nova versão do MSX HELLO! e um editor Vídeo-gráfico. O resto será surpresa!

• Pudemos assistir, há poucas semanas em São Paulo, o lançamento oficial de um kit que transforma o MSX nacional em MSX 2.0 com uma infinidade de recursos que supera a maioria dos microcomputadores mais avançados. Podemos dizer que o MSX 2.0 já chegou ao Brasil?

Eu acho que poderemos dizer que o MSX 2.0 chegou ao Brasil quando tivermos um parque instalado de, ao menos, 10 mil máquinas deste tipo.

Sem dúvida, o lançamento do kit e as transformações executadas pela empresa MSX Projetos foram um grande acontecimento para o mercado MSX. Tenho algumas máquinas adaptadas e elas funcionam perfeitamente com o software de última geração. Sem dúvida, este é o nosso futuro se nenhuma das grandes empresas tomarem uma atitude e lançaram o MSX de segunda geração em nosso mercado.

• Futuro. O que um usuário brasileiro de MSX irá encontrar pela frente?

Talvez um MSX 2, um MSX 2 PLUS, um MSX 3... Não importa a versão, o MSX tem ainda um futuro promissor no Brasil. Como dificilmente uma nova máquina poderia ser criada no Brasil, restanos observar o que acontece no exterior e, se depender disso, o MSX continua com força total, pois possui muitas vantagens sobre os AMIGA, ATARI, AMSTRAD e outros de seu nível.

CPU CARTAS

Venho acompanhando, através dos números 7, 8 e 9 da revista CPU, a implementação dos projetos MSX DEBUG e SCREEN IV. Montei o MSX DEBUG sem problemas, até o ponto em que o interrompeu no número 9 da citada revista, mas o mesmo não aconteceu com SCREEN IV.

Os erros que estão aparecendo depois de digitados os três blocos já publicados (venho informar que a soma do bloco publicado na revista número 9 está correta, e que o primeiro bloco, publicado na revista número 8, já foi revisto diversas vezes, parecendo estar sem erros), são os seguintes:

1 – Das listagens para teste publicadas na página 40 da revista número 9, nenhuma funciona. A nova tela parece ser ativada, mas nada aparece escrito nela, não parecendo ativo o comando PRINT. O comando LINE parece ser o único dos testados que está funcionando.

2 – O POKE no endereço &HF3B0, se o argumento for maior do que 40, desarruma a tela, não funcionando como substituto do comando WIDTH.

Tenho ainda as seguintes dúvidas:

1 – O endereço da rotina principal do módulo deve ser digitado em 4109H ou em 410AH? No primeiro existe o byte 0C3H, que me parece ser um delimitador dos endereços das rotinas chamadas pelo programa. Tentei as duas soluções, mas os erros acima continuaram acontecendo.

Qual é o comando correto para ativação da nova tela de 64 colunas? SCREEN IV OU SCREEN 4? Ambos parecem ser aceitos pelo programa. Com a utilização do primeiro as listagens de teste da revista número 9, o comando PRINT funciona e o LINE é recusado, mas a tela de 64 colunas não é ativada, ao que parece.

Certo da compreensão de vocês para estas minhas dúvidas, subscrevo-me,

Milton José Pinto
R. Gal. Venâncio Flores 389/204
Leblon
22441 – Rio de Janeiro – RJ

Existem algumas dúvidas em relação ao projeto SCREEN IV que pretendo esclarecer e que foram geradas pela confusão em relação ao nome do programa. Na primeira parte do SCREEN IV, dissemos que o comando que realmente habilita a tela é SCREEN 4. Como bons entendedores de BASIC, podemos perceber que o IV é um parâmetro e, portanto, uma variável. Para que o comando funcionasse, seria necessário inicializar a variável com valor correto previamente.

O leitor Milton José Pinto deve ter recaído neste problema. Quando disse que o PRINT é aceito, mas o LINE não, é porque a tela que foi ativada é a tela 0. No caso contrário, quando somente o LINE foi aceito, é sinal que a tela certa foi ativada, ou seja, a tela 4. Não é normal a rejeição do PRINT nestas condições. Espero que não tenha esquecido de incluir o novo set caracteres nesta parte, pois deste modo não deverá aparecer nada mesmo.

Outra causa pode ter sido a chamada do PRINT. O BYTE 0C3H a que se refere, localizado no endereço 4109H não é um byte delimitador, mas apenas o código da instrução de desvio JUMP do Z-80. Os próximos 2 BYTES definem o destino do JUMP. Portanto, os BYTES que definem o endereço devem ser colocados no endereço 410AH. Mais precisamente, o BYTE 86H no endereço 410AH e o BYTE 0BH no endereço 410BH. Se isso ainda não resolver os problemas, verifique os passos iniciais, principalmente a definição da chamada de entrada das rotinas e as definições dos ganchos.

É fácil explicar o fato do comando POKE ter bagunçado a sua tela. Os leitores foram advertidos que a tentativa de usar um número de colunas maior que o permitido poderia causar distúrbios. Isto deve ser considerado também para os outros modos de tela. Antes de mudar o valor para mais de 40, certifiquem-se que a tela 4 está realmente ativada.

Como último lembrete, a tela 4 ainda é uma tela instável, ou seja, só dura enquanto estiver executando um programa, assim como as telas gráficas 2 e 3.

Sérgio Duric Calheiros

Parabéns pelo sucesso que essa revista está fazendo. Mas um defeito, ou melhor, dois, devem ser citados: a revista poderia ser colorida e padronizar as letras das listagens dos programas, pois é difícil digitar com uma letra minúscula.

Apesar destes dois defeitos, considero CPU a melhor revista para MSX atualmente.

Gostaria que vocês me tirassem algumas dúvidas:

1 – Digitei 3 vezes o programa de conversão de telas para SCR e, no Graphos III, ela saía com defeitos (desnivelada, com pontos brancos, etc). Há algum defeito no programa da revista.

2 – Meu programa Screen 4 imprime na tela gráficos, mas não imprime letras. Há um erro de edição? Caso não, onde devo procurar meu erro?

3 – Onde posso comprar o copiador Prokit Zapper 1.0? Qual o seu preço?

Uma sugestão para os programadores de CPU é a de transformar os jogos MEGARAM em jogos que possam ser carregados aos poucos como o Nemesis.

Finalizando, agradeço pelas respostas e continuem fazendo o sucesso atual.

Célio Wakamatsu
R. Albuquerque Lins 772 aptº 101
Higienópolis
01230 – São Paulo – SP

Com relação à sua dúvida do programa SCREEN 4, solicitamos que leia a resposta que foi dada ao leitor Milton José Pinto. Caso ainda fique em dúvida, não hesite em nos escrever novamente.

O programa conversão de telas para SCR tem gerado várias dúvidas por parte dos leitores e já estamos com uma nova versão do programa, bem mais completa e fácil de ser digitada e utilizada, que será publicada em um dos próximos números de CPU, pois foi um programa que despertou um grande interesse.

O programa Prokit poderá ser encontrado em várias softhouses, que possuam a representação para revendo do produto. Consulte os nossos anunciantes.

Estou lhes escrevendo porque gostaria que o mapa do jogo Fredy Hardest II fosse publicado nas próximas edições.

As senhas do jogo Goonies são:
2º fase – Sr. Sloth
3º fase – O corsário
4º fase – Fratelli
5º fase – Willy

Cristiano Jannone Carion
Rua Des. Coufal 596
B. Ipanema
91900 – Porto Alegre – RS

Possuo um Hotbit e peço que fossem publicadas dicas, um mapa e o que você tem que fazer quando entra em um computador do jogo Navy Moves 2.

Fernando Baibich
R. XV de Novembro 1517 ap 704
80060 – Curitiba – Paraná

O jogo Navy Moves teve o seu mapa publicado em CPU número 8, mas mais dicas virão, é só aguardar.

Gostaria de parabenizá-los pela qualidade de CPU e, dando uma sugestão, queria que vocês colocassem uma seção de dicas, com programas de pequeno porte, além de uma matéria sobre o jogo FEUD, que é de difícil conclusão.

Para quem estiver interessado, gostaria de corresponder-me com usuários da linha MSX, para troca de dicas e programas em Basic e Assembler.

Juliano R. C. Mello
Rua Panorama 574
Palmas do Iremenmbe
02347 – São Paulo – SP

Antes de tudo, gostaria de parabenizá-los pela revista. Acho que já estava mesmo na hora de termos uma revista para o MSX de nível profissional, com uma linguagem acessível.

Pena que não posso dizer o mesmo sobre "drives" que temos, e é sobre eles que quero discutir.

Primeiro comprei um TPX que, após 8 meses, deu defeito. Fui à loja que o vendeu e fiquei sabendo que o TPX não estava mais sendo fabricado, começando então, uma peregrinação, por 3 "empresas" de conserto, indicadas pelo vendedor. Depois de 2 meses de vai e volta, meu "drive" foi dado como inconsertável.

Comprei um DDX que, exatamente 2 dias depois de vencida sua garantia de 3 meses, começou a dar erro de leitura. Pronto! Começou tudo de novo. Só depois de rodar muito, consegui descobrir uma "empresa" que o aceitasse. Durante este "round" também achei uma outra loja que aceitou o desacreditado TPX.

Pasmem! Em uma semana o TPX, consertado numa loja que foi descoberta ao acaso, ficou pronto e passa bem. Já o DDX, mais novo está na "via crucis" há 5 meses!

Creio que os ditos "fabricantes" de drives nacionais (que vocês sabem que é tudo importado) precisam, urgentemente, de um chá de profissionalismo, melhorando a qualidade dos produtos e de sua assistência técnica, principalmente pelo preço que o consumidor paga pelos aparelhos.

E por falar em preço, um MSX completo (CPU, teclado, monitor, 2 drives e expansão para 80 colunas) não sai por 1/5 do preço de um 16 bites. O XTEC, por exemplo, sai, completo na faixa de NCz$ 4.700,00, enquanto que um Expert, em vez de NCz$ 940,00, custa NCz$ 3.100,00, ou seja, aproximadamente 2/3 do preço do de 16 bites.

Sem mais, espero que os dois tópicos aqui (preço e qualidade dos produtos) sejam levados aos demais leitores e aos "fabricantes" de drives.

Antoni Avila
R. Belfort Roxo 351/1201
Copacabana
22020 – Rio de Janeiro – RJ

Realmente o problema da qualidade dos equipamentos de informática nacionais e da assistência técnica é um problema sério no mercado brasileiro.

CPU CARTAS

Os fabricantes, escorados por uma Lei de Informática que os protege, não levam a questão da qualidade dos equipamentos muito a sério. Para piorar a situação, as assistências técnicas prestam um serviço deficitário, sem garantia e qualidade. É claro que existem exceções e, procurando, podemos encontrar um ou outro fabricante ou assistência técnica que estão sabendo trabalhar e que não precisam do chá de profissionalismo que você falou.

Infelizmente, também compramos um drive da DMX, acompanhado de fonte e interface, que nunca conseguiu formatar um disquete ou efetuar a gravação de um arquivo, sendo usado somente para leitura. O fabricante dos produtos DMX não se preocupa com o usuário, pois sequer coloca nos seus produtos o seu endereço ou telefone para contato e, muito menos, uma relação de assistências técnicas, sendo o ilustre desconhecido, que quer vender o seu produto e sumir.

Como o usuário não tem muitas opções, acaba ficando na mão de poucos fabricantes de equipamentos. Contudo, temos a certeza que esta situação irá mudar e nós, da revista, iremos trabalhar no sentido que isso possa ocorrer rapidamente, para que o usuário possa usar seus equipamentos sem dores de cabeça, como acontece quando se compra uma máquina de lavar, por exemplo, onde o fabricante não se esconde e presta uma total e excelente assistência ao usuário, pois quem já comprou um equipamento de um fabricante qualquer é o potencial comprador de outro mais moderno, quando for lançado, desde que o que tenha sido vendido seja de qualidade.

Fiquei satisfeito, ao abrir a revista CPU numero 7 e verificar que agora temos uma revista que publica programas mais úteis dos típicos círculos na SCREEN 2 ou corrida do ratinho.

Gostei muito também das rotinas gráficas para o Pascal e espero que vocês dêem continuidade àquela matéria, com novas rotinas, como, por exemplo, uma rotina para salvar telas gráficas.

Outro ponto que gostei muito da revista foram as dicas de utilização de utilitários, como o M80 e o Bascon. Muitas vezes, são publicados programas que utilizam softs como estes, mas os programadores com menos experiência ficam apenas com água na boca, pois não têm a mínima idéia de como rodar os programas.

Eu possuo um Expert da Gradiente transformado em 2.0 e gostaria, como muitos que já o possuem, que aparecesse, de vez em quando, matérias e programas para esta versão, bem como se observasse, em programas em assembler, a compatibilidade (uso da memória).

Gostaria, se possível, que se publicasse uma procedure em Pascal para acessar as SCREENS do 2.0, bem como um programa para DUMP de telas gráficas 5, 6,7 e 8, em impressora. Não sei se vocês possuem algum micro 2.0 ou se tem condições de publicar tais artigos, mas, de qualquer maneira, fica a sugestão.

Estão de parabéns, vocês da CPU, e espero que continuem cada vez melhor.

Marcus G. de Amorim
HCGN 707 bl E casa 19
70740 – Brasília – DF

Novas rotinas para o Pascal serão publicadas, é só aguardar.

Nos próximos números de CPU, estaremos dando maior destaque ao MSX 2.0, pois a nova versão do MSX começa a ganhar força no Brasil e cada vez é maior o número de usuários que possuem micro transformado.

Nosso MSX 2.0 transformado irá chegar em breve e, logo que isso ocorra, nossos colaboradores começarão a desenvolver programas, a exemplo do que vem acontecendo com o MSX 1.

Se você sabe programar, porque não manda suas colaborações para a revista? Teremos a maior satisfação em analisar suas colaborações, sendo que artigos publicados são remunerados.

Eu sou um novo assinante da revista CPU. Assim que vi a revista no jornaleiro, logo comprei e assinei, pois gostei muito e queria parabenizá-los por esse belo trabalho.

Gostaria de esclarecer algumas dúvidas sobre o Expert:

1 – Como faço para editar o list em meus programas?

2 – Como posso fazer com que o programa não pare quando receber as ordens das teclas "CONTROL+STOP"?

3 – Como se usa o Modem?

4 – Eu posso ligar qualquer máquina de escrever elétrica no computador para que ela funcione como impressora? Qual o nome do periférico dessa conexão? Onde posso encontrá-lo, aqui no Rio.

Agradeço a compreensão de vocês.

Rodrigo Machadi Valladão
Av. Ary Parreiras 74/304
24230 – Niterói – RJ

Para travar o list, basta dar o seguinte comando: POKE &HFF89,&HE1 (para que o computador acuse Sintax Error quando se solicitar o List), ou POKE &HFF89,&HE5 (para que o computador reset quando for solicitado o List). Para que o computador volte a aceitar o List, comande: POKE &HFF89,201.

O Control Stop pode ser desativado com POKE &HFBB1,1 e ativado novamente com POKE &HFBB1,0.

Um dos próximos números de CPU será dedicado exclusivamente a modem e transmissões de dados.

Existem máquinas elétricas que podem ser adaptadas para serem utilizadas como impressoras, como a Praxis, da Olivetti. Para conectar uma máquina de escrever elétrica a um computador, é necessário ter uma interface paralela.

No caso da Praxis, esta interface pode ser adquirida na Mesbla e instalada pela própria Assistência Técnica da Olivetti.

Gostaria que vocês publicassem o mapa e dicas sobre o jogo "La Abadia Del Crimem" e me indicassem o que fazer com o jogo Br-116, publicado na edição 5, já que, depois de digitado e conferido, ao rodá-lo, apresenta erro de sintaxe na linha 130.

Allessander Christian Miranda
Av. Braz Leme 2241
Santana
02022 – São Paulo – SP

O mapa do jogo Abadia Del Crimem foi publicado em CPU número 9.

A listagem do jogo BR-116 está correta e não encontramos nenhum erro de sintaxe na linha a que se referiu. Verifique mais uma vez toda a listagem, principalmente os caracteres especiais, que as vezes são de difícil identificação.

Logo que comprei a revista CPU número 8, vi o programa do jogo Atlântida MSX e o digitei, mas, ao executá-lo, o programa não funcionou. Gostaria de saber se o programa tem algum erro ou se, ao digitá-lo, errei alguma coisa.

Gostaria de agradecer a publicação dos mapas dos jogos Titanic I e II e Terra Mex, pois com esses mapas e as dicas terminei os três jogos.

Gostei também da publicação dos mapas e macetes do jogo La Abadia Del Crimem.

A partir do número 9, assinei a revista e já recebi este exemplar. Não sei porquê ele chegou em minha casa após ter chegado nas bancas de Belo Horizonte.

Lendo a seção CPU Cartas, verifiquei que há muitas pessoas interessadas em códigos, Pokes e macetes de jogos, assim como eu.

Acho que posso ajudar e ser ajudado por vocês. Tenho senhas do jogo Goonies e do Jogo Venon Strikes Back, que são:

Goonies: MR Sloth e One Eyed Willy

Vennon Strikes Back: Mayhem, Transmogrify e Valkyr.

Gostaria de receber informações como Pokes, macetes e mapas. Seria muito bom, por exemplo, mapas dos jogos Black Beard e Avenger, pois gosto muito desses jogos e outros desse tipo, mas não consigo jogá-los.

Conheço também, macetes para alguns jogos. De todos eles, o mais interessante é o Poke de vidas infinitas para o Batman.

Pretendo, posteriormente, adquirir os números de 1 a 7 da revista CPU.

Humberto Miranda de Andrade
Av. Prudente de Morais 1965/302
Cidade Jardim
30380 – Belo Horizonte – MG

Suas dicas sobre jogos ou qualquer assunto relacionado a MSX são sempre bem vindas e, se forem publicadas na revista serão remuneradas.

Envie-nos logo seus macetes e seja também um dos colaboradores de CPU.

Tenho uma pequena falha a apontar no artigo "Cursor piscante", da edição de junho. A listagem em Basic do programa, na linha 140, necessita do complemento ":NEXT I" para funcionar corretamente.

Gostaria que vocês publicassem mais Pokes para jogos.

Sérgio Luiz da Silveira
Estr. V. da Pavuna 4800 bl 12 apto 308
20760 – Rio de Janeiro – RJ

Possuo alguns programas de minha autoria e gostaria de saber como colaboarar com esta conceituada revista, através da publicação dos mesmos, caso houvesse interesse. Existe alguma forma de incentivo?

Gostaria que meu endereço fosse publicado para a troca de idéias e programas com outros leitores de CPU.

Renato Paulo de Mello
Rua Tuiuti 2350
Bairro Aventureiro
89200 – Joinville – SC

Todas as colaborações enviadas para CPU são bem recebidas e, se publicadas, são remuneradas, conforme a avaliação de nossa equipe. Antes de efetuarmos a publicação entraremos em contato, informando o valor da remuneração bem como para solicitar que nos seja enviada autorização para publicação.

ANÁLISE DA MEGARAM

Divino C. R. Leitão

Quando se fala de jogos, um dos periféricos mais interessantes do MSX é a MEGARAM. Trata-se de um cartucho que acrescenta "apenas" 256 Kb de memória ao micro. É um mistério o fato de uma expansão com esta capacidade ser utilizada apenas em jogos, mas como esta edição é uma edição de jogos, vou deixar esta discussão para um próximo número.

Recebi o produto sem um manual de instruções, a única indicação de sua procedência é uma etiqueta da "CHEYENNE ADVANCED SYSTEMS". Sua comercialização é feita por diversas softhouses, sendo que a utilizada nesta análise foi fornecida pela PAULISOFT INFORMÁTICA, juntamente com 5 jogos específicos para MEGARAM.

Os jogos que exigem MEGARAM rodam em um MSX normal, porém só funcionam com o periférico instalado ao micro, suas características mais marcantes são quantidades enormes de paisagens e fases diferentes, que tornam os jogos muito mais atraentes.

Todos os jogos enviados pela PAULISOFT rodam sob o gerenciamento de um outro programa, o GAME MASTER. Como não pude encontrar referências sobre a real função do GM (GAME MASTER), meus comentários serão apenas sobre observações empíricas do mesmo.

Aparentemente, o GM executa os jogos após o carregamento destes na MEGARAM, mas não fica só nisto, permite interromper os jogos e gravá-los ou carregá-los na fase em que se encontram, gravar os recordes ou a tela do momento da paralisação. Tais recursos funcionam em disco e K7, mas houve momentos em que este procedimento ocasionou o congelamento do micro, ou seja, seu funcionamento foi muito irregular mas de qualquer forma funcionou na maioria dos testes.

Uma outra opção do GM, permite o HARDCOPY (cópia na impressora) da tela do momento da paralisação de um jogo, mas não consegui obter cópias em impressoras nacionais e nem mesmo em uma Epson original, de forma que tornou-se um recurso inútil durante os testes, mas como os testes foram efetuados em apenas três impressoras, uma Lady 80 da Elgin, uma Grafix 80 e uma Epson FX 80, não posso afirmar que tais recursos não funcionam, fica para um artigo futuro.

As gravações de tela obtidas com o GM não são compatíveis com a gravação de tela padrão MSX, daquela que pode ser carregada por BLOAD" ",S. Só consegui recuperá-las através do próprio GM, sendo que, em uma tentativa de ler o arquivo sem usar o GM, consegui ler o conteúdo do mesmo, mas os códigos não faziam sentido. Aparentemente as telas são gravadas em um modo compactado ou codificado.

Os recursos do GM não se revelaram de grande importância, a não ser pela possibilidade de gravar um jogo na fase atingida para poder continuá-lo posteriormente. Talvez com alguma documentação sobre o mesmo se possa usar mais seus recursos. Deixo a peteca com vocês, fabricantes e distribuidores.

Mas vamos à parte realmente quente da MEGARAM, e objetivo desta análise, que são os jogos propriamente ditos. Foram cinco os programas analisados, mas existem muitos outros disponíveis nos catálogos das softhouses e também nas revistas estrangeiras.

PENGUIM ADVENTURE

Quem já viu o Antartic Adventure e se emocionou com o simpático pingüim da Konami, certamente vai amar este megagame, que é uma continuação muito melhorada do antecessor. Só consegui atingir até a quinta fase e as surpresas sucediam-se. São muitas as novidades. Os peixes, que no jogo anterior apenas contavam pontos, agora servem como dinheiro para comprar objetos que aumentam a capacidade de movimentação do pingüim. Monstros gigantes, uma gostosa máquina caça-níqueis e locais mirabolantes, tornam este jogo uma agradável diversão para adultos e crianças. É um jogo para ser jogado e assistido. Os locais de movimentação são muito mais variados e alguns inusitados, como, por exemplo, no espaço sideral entre meteoros gigantes e peixes voando em pleno espaço. Alguns dos ambientes são cavernas, rios, e fundo do mar.

Toda esta variedade e uma grande dose de criatividade e graça nos desenhos fizeram com que eu elegesse o Pingüim como o melhor jogo. É claro que os outros também são fantásticos, mas a simpatia do pingüim me cativou e também ao meu sobrinho de 8 anos, o Rafael, que, se deixar, quer passar a noite inteira jogando. Em tempo, ele conseguiu quebrar meu recorde, quando ambos estávamos frios no jogo.

SPIRIT

Uma novidade em termos de jogos com automóveis é o que se revelou o Spirit. A parte relativa ao jogo não tem grandes novidades, excetuando-se uma movimentação bem próxima da realidade, com derrapagens, capotagens e outros efeitos que vão depender do modo de jogo escolhido.

A novidade fica por conta da escolha do carro, você pode literalmente construir seu bólido, seja de F1 ou Rally, especificando chassis, motor e potência do carro, com o auxílio de ótimas gráficos e informações técnicas sobre os carros.

Honestamente, achei-o um jogo um pouco cansativo, mas que vai agradar os aficcionados de corrida, pois possui diversos recursos e infindáveis pistas, com

destaque para a exatidão dos traçados das pistas de F1, onde está incluído o circuito de Jacarepaguá (ou Nelson Piquet) do Rio de Janeiro.

NEMESIS III

A coqueluche dos jogos tipo penetrator já está na terceira versão e mais emocionante do que nunca. Gráficos belíssimos e animações surpreendentes irão fazer a alegria dos fãs das naves Nemesis e seus infindáveis inimigos. As paisagens de fundo são espetaculares, assim como os obstáculos a serem vencidos.

O jogo não traz modificações estruturais em relação aos anteriores, mantendo o mesmo padrão de qualidade que tornou o Nemesis um clássico dos penetrators e permitindo uma rica coleção de novidades, devido à grande capacidade de memória conseguida com a placa. É um jogo que o possuidor da MEGARAM não pode dispensar.

GALL FORCE

Depois de GALAGA e ZANAC, GALL FORCE demonstra ser o máximo em termos de jogo tipo "shotting-in-up". Uma grande variedade de naves e inimigos fazem deste jogo uma distração sem fim. Os tipos de armas são os mais variados, assim como as naves usadas, cada qual com seu piloto, todos com aquela carinha de "japonês punk", bem no estilo dos heróis que fazem a alegria da garotada no vídeo.

É muito fácil passar horas jogando Gall Force sem perceber. O jogo parece estar sempre recomeçando, tal a variedade de paisagens e objetos que surgem na tela comos mais incríveis movimentos. Também é uma opção imperdível para os amantes do gênero.

KING'S VALLEY 2

A maldição das pirâmides continua e agora você tem nada mais nada menos do que 60 salas a explorar. E cada uma é um verdadeiro quebra-cuca, pois o King's Valley é um tipo de jogo que envolve um certo grau de raciocínio para se poder vencer cada fase.

Não é necessário dizer novamente que os gráficos são belíssimos, sendo que algumas das paisagens dão a sensação de que a resolução do MSX 1 foi ampliada, mas não é nada disso, é um trabalho de profissionais da animação que brindam os felizes possuidores de MEGARAM.

Foram apenas cinco jogos para serem analisados. Não são os melhores da MEGARAM e talvez não o sejam nem do MSX, mas sem dúvida são programas de excelente qualidade, sadios e que certamente irão proporcionar aos seus usuários muitos e muitos momentos de descontração e lazer.

Durante todo o tempo, elogiei os gráficos excelentes e a quantidade de opções de cada jogo, mas os jogos MEGARAM tem ainda um trunfo a mais que os jogos comuns, o som, as trilhas sonoras e efeitos especiais, tais como tiros, ruídos e explosões, todos são de altíssimo nível o que torna perfeita a integração do jogador com o ambiente do jogo.

CONCLUSÃO

Apesar de ser um periférico caro e não ter ainda outras opções além dos jogos, e isto nós vamos discutir no futuro, revelou-se uma excelente aquisição, pois ampliou o campo de uso dos MSX brasileiros que padecem atualmente do mesmo mal que todos os micros em nosso território. Resta esperar que a MEGARAM possa ser produzida a um preço mais acessível que os 100 dólares que custa atualmente e, principalmente, que a sua capacidade venha a ser aproveitada também para programas utilitários, tais como banco de dados, planilhas e editores de texto, que, com 256 Kb de memória, tornariam o MSX uma máquina muito mais profissional, que de "quebra" teria joguinhos fantásticos.

Voltarei ainda a falar deste periférico, abordando suas outras características e possibilidades. Nesta edição a MEGARAM fez o vestibular dos jogos, e passou com louvor.

MSX DEBUG

Sérgio Duric Calheiros

Até a parte anterior do projeto MSXDEBUG, tivemos uma visão geral da organização do programa e de como foi construído. Sabemos como o programa reconhece comandos, como tratar os parâmetros e como implementar novas instruções.

Como todo programa do porte do MSXDEBUG, este também não está livre de erros de programação. São pequenos erros que, apesar de não atrapalharem o funcionamento normal do programa, incomodam, porque existem. Um erro pode ser definido como algo que foge à especificação inicial do programa. Uma deficiência é algo que funciona, mas não como deveria funcionar. Trataremos dos dois sempre que necessário. Ao invés de publicarmos notas de correção ou simplesmente ignorá-lo, decidi fazer uso disso para mostrar como usar o próprio MSXDEBUG para nos auxiliar a contornar estes tipos de problemas e, também mostrar em que tipo de armadilhas estamos sujeitos a cair.

Um desses problemas está no comando DISP. Como vimos em artigos anteriores, existe uma rotina no MSXDEBUG, encarregada de converter um dado numérico presente num buffer, chamada @CONVD.

Caso haja algum digito ilegal neste buffer, ou seja, fora do intervalo permitido para um número hexadecimal, ela nos manda uma mensagem de erro. Para que isso seja evitado, devemos evitar mandar digitos ilegais. No comando DISP, a rotina que se encarrega de filtrar os dígitos ilegais não filtra os dígitos entre) e A. Se algum desses caracteres for teclado acidentalmente, apesar de não causar nenhum dano à memória ou perda de dados, há interrupção do comando com a mensagem de erro.

Na listagem 1 está a rotina de filtragem, tal como foi criada inicialmente. Seu funcionamento se resume na verificação do intervalo do dígito na tabela ASCII. Como em assembler não temos recursos tais como IF x<a AND x>b THEN do Basic, temos que simular essa instrução de alguma maneira. A única ferramenta que dispomos para isso são as flags Z-80, as instruções comparação e as instruções de desvio.

Para comparar valores, usamos a instrução ComPare do Z-80. Esta instrução sempre usa o registrador Acumulador como alvo e sempre altera todas as flags. Para comparar valores iguais, devemos verificar o resultado na flag de Zero, ou seja, a instrução que segue o ComPare deve ser uma instrução de desvio que use esta flag tal como JR Z, CALL Z, RET Z ou aquelas que verificam a desigualdade como JR NZ, CALL NZ e RET NZ.

Se precisarmos comparar se um valor é maior, menor, maior ou menor igual que outro, devemos verificar o resultado da comparação na flag Carry.

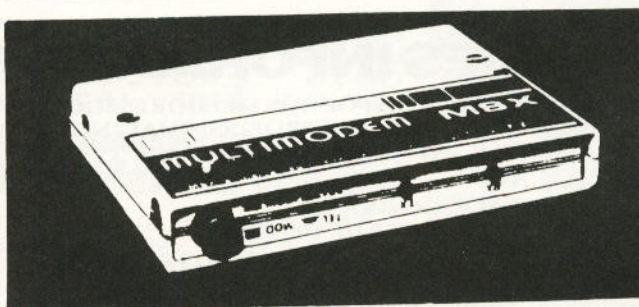
Queremos verificar se o valor presente no Acumulador é 7DH, por exemplo, basta usar a instrução CP 7DH. Se o valor do Acumulador for igual, então Z=1, senão Z=0. Se o valor for menor que o 7DH, então C=1 e se o valor for maior ou igual, então C=0.

Não há maneira de verificar, diretamentem, se o valor do Acumulador é "maior que" ou "menor igual" a algum outro valor. Portanto, devemos lançar mão de outra instrução ComPare que, no caso, usa registradores no lugar de constantes. Para isso, basta colocar o valor do Acumulador em algum registrador que seja conveniente, em seguida colocar o valor que gostariamos de comparar com o próprio Acumulador e, finalmente, usar o ComPare.

Isto está exemplificado na listagem 1.

O intervalo permitido está entre "0" e "9", entre "A" e "F" e entre "a" e "f".

Examinando a rotina, podemos entender como ela funciona. Em primeiro lugar, verifica-se se o valor é menor que "0". Se for, a rotina encerra com a flag de Carry "setada", indicando um caracter ilegal. O próximo passo é verificar se é maior que "f", que é o limite superior do intervalo. Uma letra minúscula tem o código ASCII maior que sua minúscula e a diferença entre seus valores é 20H ou 32D. Em seguida, a rotina verifica se o caracter é menor que "a", com possibilidades de ser letra ou alga-

rismo. Se for menor, o dígito está entre "a" e "f", sendo necessário convertê-lo para seu equivalente maiúsculo. Sendo menor que "a" o dígito também deve ser menor que "F", pois este caracter passa a ser o limite superior a partir deste ponto.

Quem chegou até aqui sem dificuldade, já deve ter percebido onde está o erro. A próxima comparação deve verificar se o caracter está abaixo de ou igual a "9" e não abaixo de "a". Depois, verificar se é a maior ou igual a "A". Portanto, as últimas instruções da rotina devem ser como está na listagem 2.

Resumindo: o erro esté em um único BYTE da rotina.

Para consertar, carregue o MSXDEBUG a partir do endereço 4100H e troque o BYTE 61H, que está no endereço 44E4H, pelo BYTE 3AH. Aproveite para colocar o BYTE 6 1H no endereço 41ECH. Isto evitará que as teclas de função sejam apresentadas quando saimos do comando. Salve o programa, rode e teste-o.

Se ficou alguma dúvida, escrevam. No próximo mês, implementaremos um novo comando. É um comando de rastreamento de caracteres na memória, chamada BUSCA. Quem quiser, já pode preparar seu reconhecimento e a chamada. Posso adiantar que o endereço inicial da rotina @BUSCA é o final da @SOMAR. Não esqueçam de colocar a instrução de retorno (0C9H) neste endereço

```
LISTAGEM 1

ALT35:   CP   30H      ;Compara o Acumulador com 30H
         RET  C        ;Se A<30H (C=1) sai sem aceitar digito
         LD   C,A      ;Prepara comparacao tipo "maior que"
         LD   A,"f"    ;Coloca limite superior em A
         CP   C        ;Compara o valor anterior com "f"
         LD   A,C      ;Recupera valor de A
         RET  C        ;Se A."f" (C=1) entao sai sem aceitar
         CP   "a"      ;Verifica se A<"a"
         JR   C,ALT36  ;Se for, deve ser maiuscula e desvia
         SUB  32       ;Senao, muda minuscula para maiuscula
ALT36:   LD   C,A      ;Repete comparacao para A>"F"
         LD   A,"F"    ;
         CP   C        ;
         LD   A,C      ;
         RET  C        ;Se A>"F", sai sem aceitar
         CP   "a"      ;Se for menor que "A" sai sem aceitar
         RET  C
ALT37:   AND  A        ;Sinaliza que o caracter e legal com
         RET           ;a flag de Carry ressetada.


Listagem 2

         RET  C        ;Se A>"F" sai sem aceitar
         CP   "9"+1    ;Compara com ":" ("9"+1)
         JR   C,ALT37  ;Se for menor sai como legal
         CP   "A"      ;Compara com "A"
         RET  C        ;Se for menor sai como ilegal
ALT37:   AND  A        ;Sinaliza que o caracter e legal com
         RET           ;a flag de Carry "resetada"
```

SCREEN IV

Sérgio Duric Calheiros

Nas primeiras partes do projeto SCREEN IV implementamos os módulos responsáveis pelo gerenciamento geral das rotinas e pela geração de caracteres em 64 colunas na screen 4. Nesta parte, serão implementadas as rotinas que ajustam o comando WIDTH para comportar o novo número de colunas.

Antes de começar a descrever os passos da implementação, seria interessante deixar o leitor a par da programação estabelecida para o projeto.

Todos os módulos do SCREEN IV são interdependentes, ou seja, a implementação de uma parte do módulo não pode ser feita sem a anterior. O SCREEN IV só estará completo em si quando implementarmos a parte que inclui as rotinas do editor BASIC. Antes disso, devemos implementar as rotinas de apoio ao ambiente novo que criaremos mais tarde. Após a conclusão do novo ambiente encontrado na screen 4, começaremos a estender o BASIC com os novos comandos.

Seguindo esta linha, os próximos passos que sucederão a implementação do comando WIDTH deste mês será a ativação do cursor e a habilitação das teclas de função. Finalmente, implementaremos o módulo mais importante para trabalhar com a SCREEN IV. Será o editor-interpretador das instruções do BASIC diretamente do teclado na própria screen 4. Após isso, restam os novos comandos, como o LIST melhorado, o comando FILL, o comando OVER, os comandos SCREEN, CLS ON e CLS OPE. Além disso, ainda resta PRINT @, um comando que, praticamente, deixará o BASIC do MSX quase que totalmente compatível com o BASIC do velho conhecido TRS-80.

Como podemos perceber, o ritmo do projeto foi um pouco desacelerado, mais por necessidade que por opção. A implantação do comando WIDTH é rápida e sem maiores problemas. Se você ainda não consegue digitar os módulos anteriores, pode começar sem presssa.

Quanto à implantação do WIDTH, siga os passos de sempre. Para quem usa o MSXDEBUG ou algum outro utilitário, limpe a área da página 1 (endereços 4000H a 7FFFH). Carregue a última versão do SCREEN.COM, com todos os módulos anteriores, no endereço 4100H. Comece a digitar o bloco 1 a partir do endereço 4EF7H. A partir deste endereço só deve haver BYTE 00H. Terminada a digitação do bloco, devemos fazer com que o comando seja reconhecido pelo módulo de gerenciamento das rotinas. Então, no endereço 410DH, coloque o valor 0EF7H, que é o endereço de entrada do comando WINDTH. Não se esqueça que a parte menos significativa vwm primeiro, ou seja, o BYTE 0F7H dever ser colocado no endereço 4100DH e o BYTE 0EH no endereço 410EH. Agora, devemos habilitar a chamada do comando que está na ROM. No endereço 4167H, coloque o valor 0FF84H, sempre com a parte menos significativa primeiro.

PAULISOFT
Compare!
MSX
MSX
Software 100% nacional, desenvolvido pela Paulisoft com manual, cópias com nº de série, garantia de up to date e assistência ao usuário.
FAST COPY
Para a vergonha dos micros de 16 bits e muitos Kbs de memória.
Copia um disco completo no seu MSX mais rápido que num PC. Precisa dizer mais alguma coisa?
Copiador de discos ultra-rápido para controladoras Padrão Microsol.
EDTRONIC
EDTRONIC
VER. 1.1 - AGO/88
Autor: PAULO M. FIGUEIRA
PAULISOFT Informática Ltda
SERIAL #: 8998
Finalmente alguém pensou em você, técnico ou hobbista de eletrônica, e criou um auxiliar para seus projetos.
Tabela padrão de simbologia em eletrônica; Recursos para edição, Montagem e impressão de esquemas para projetos eletrônicos.
Acompanha arquivo exemplo.
MSX TURBO
MSX-Turbo
(c) 1988 Paulisoft Informática
Caixa Postal 8909
02221 São Paulo SP
SPRITE MAKER
SPRITE MAKER®
AUTOR: FABIO A. R. CORREA
PAULISOFT SOFTWARE
CAIXA POSTAL 64019
CEP 02227 - São Paulo SP
Super editor de sprites 16x16 que inclui rotinas para reversão, espelho de 1/2 e 1/4.
Compatível com o Graphic View.
Acompanha super manual.
Não é mágica, é tecnologia!!! Um incrível software que vai deixar suas rotinas de cálculo e plotagem de gráficos de 6 a 20 vezes mais rápidas!
MSX TURBO é um compilador que opera na memória, acelerando incrivelmente as operações de cálculo.
Um genial programa para incrementar em suas telas gráficas rotinas de Scroll (movimentação de telas) selecionadas, a fim de que com facilidade você possa criar um SHOW VISUAL.
GRAPHIC VIEW
GraphicView
PRODUTOS ORIGINAIS COM TOTAL GARANTIA • PEÇA CATALOGO COMPLETO COM NOSSOS PRODUTOS INTEIRAMENTE GRÁTIS
Um novo Conceito em Soft
Também nas melhores lojas e softhouses do Brasil
PROGRAMAS APLICATIVOS/UTILITÁRIOS
KIT PAGE MAKER • MSX PAGE MAKER • MSX PORTIFOLIO • MSX CHART • HELLO • MSX DESIGNER • MSX VIDEO GRAPHICS PLUS • FLUXO DE CAIXA • EDARQ • VOX • LENDA DA GAVEA • AMAZONIA • SERRA PELADA • LINHA PRO KIT • GRAPHOS III • DBASE II PLUS • SUPERCALC2.
Aguarde!
BKP - Disco: por Júlio Veloso, com novas versões e novos comandos implementados.
Na Paulisoft você encontra: Drives; Expansão 256 Kb; Disquetes; Fita de video MPO; Livros Aleph e muito mais.
Envie seu pedido para Caixa Postal 64.019 CEP 02227 - São Paulo - SP
Se você mora em São Paulo, faça-nos uma visita.
PAULISOFT
Rua Coronel Xavier de Toledo 1231 Cj. 31/32 (a 20 metros da estação do metrô Anhangabaú) CEP 01048 - São Paulo - SP
Seu MSX precisa nos conhecer!

Como último procedimento, salve o bloco. Se você ainda não sabe como salvar, desista. Mãs não se esqueça dos endereços! Caso queira, verifique a soma do bloco para se certificar que os dados foram digitados sem erros.

Com tudo certo, saia para o DOS e execute o SCREEN.COM.

Estando no BASIC, digite o programinha de teste que está na listagem 1. Como esperávamos, o comando WIDTH deve ser comportar o novo intervalo de parâmetros até 64, mas só até 64. Se outro valor fora do intervalo for tentado, é lógico esperar uma mensagem de erro.

O comando WIDTH foi implementado. Nós próximos números, este comando será complementado. Não que esteja incompleto, mas atuará em conjunto com os comandos CLS e CLS OFF.

Por enquanto, isto é tudo. Até mês que vem!

```
BLOCO 1

4EF7 CD FD 0E C3 A0 02 A7 20
4EFF 07 DD 21 5A 47 CD 7B 02
4F07 DD 21 0D 52 FE 41 30 F1
4F0F 3A B0 F3 BB C8 7B 32 B0
4F17 F3 32 6C 38 7B D6 0E 30
4F1F FC C6 1C 2F 3C 83 32 B2
4F27 F3 3A 4D 38 A7 C8 E5 CD
4F2F C6 02 E1 DD 21 DE 08 C9

Soma total:001F9F
```

LISTAGEM 1

```
10 REM Testa novo comando WIDTH
20 SCREEN 4:REM Habilita tela gráfica e texto em 64 colunas
30 PRINT "Temos agora 60 colunas para texto"
40 PRINT "Pressione uma tecla..."
50 A$=INPUT$(1)
60 WIDTH 40
70 PRINT "Voltamos a ter somente 40 colunas"
80 PRINT "Pressione uma tecla..."
90 A$=INPUT$(1)
100 WIDTH 64
110 PRINT "Esta é a tela máxima!"
120 FILES:PRINT
130 PRINT "Prepare-se para uma mensagem de erro e tecle algo!"
140 A$=INPUT$(1)
150 WIDTH 65
```

APRENDA A FAZER JOGOS NO MSX

Divino C. R. Leitão

Os jogos de computador, sempre foram motivos de desdém por parte de uma boa parcela de usuários e programadores, dito sérios, que não podem se ocupar destas bobagens. Os mais radicais, chegam ao cúmulo de taxar alguns equipamentos de vídeo-game sofisticado, como é o caso do nosso MSX, esquecendo da importância que um vídeo-game pode representar para, por exemplo, acabar com a inibição natural que as pessoas não iniciadas, adultos e crianças, sentem com relação à informática, isto sem citar o aspecto do lazer em si, que sozinho já justifica com larga margem de vantagens a existência destas maravilhas que são os joguinhos.

Minha mãe, por exemplo, cresceu em um mundo sem computadores, nunca conseguiu entender muito bem minha paixão por eles, e sempre teve um verdadeiro pavor de ficar sozinha em uma sala com uma máquina destas presente, mesmo que desligada. Foi através de um Atari, o vídeo-game mais popular do mundo, que ela perdeu este medo e hoje coloca como prioridade o estudo da informática para minha irmã caçula, sem contar que adora jogar um River Raid no Atari.

Não pretendo fazer a defesa dos vídeo-games, eles estão muito bem sem ela, apenas queria desabafar um pouco a raiva que se acumula toda vez que ouço um metidinho qualquer dizer – Ah! é só um joguinho! – é claro que o fulano tem pilhas de disquetes com jogos e provavelmente joga escondido no banheiro, pra família não ficar sabendo, mas a aparência de usuário sério tem que ser preservada a qualquer custo, isso é o que mais me irrita.

Na área de criação, a coisa se complica mais ainda, enquanto no exterior os criadores de games são respeitados como qualquer profissional merece. Aqui, se você quiser ser expulso de uma rodinha de programadores/analisatas, é só dizer que faz ou gosta de fazer jogos. O olhar de admiração logo se transforma em um indignado muchocho e, discretamente, você se vê fora do círculo, como uma coisa esquisita.

É compreensível, enquanto praticamente qualquer um, com um pouco de estudo, pode usar programas ditos profissionais ou mesmo criá-los. Fazer jogos exige uma veia artística que não se ensina, nasce com a pessoa. É como tocar piano, qualquer um pode aprender, mas poucos se tornam um Moreira Lima, pois isto exigirá, além de uma boa dose de dedicação, um talento nato. Todos os artistas afirmam que seu trabalho é 90% suor e 10% talento. Eu até concordo mas afirmo que este talento ninguém cria, é coisa de berço.

Como você pode descobrir se tem talento para ser um game-maker? É simples. Pegue seu micro, encoste os disquetes de aplicativos e comece com um joguinho de forca. Pensa que é fácil? Pois acertou, é muito fácil mesmo, em minutos você fará um.

Não sou o Moreira Lima dos vídeo-games. Se micro fosse piano eu não tocava nem em cabaré brasileiro, mas gosto do assunto e ao longo dos anos adquiri alguns conhecimentos que gostaria de dividir com vocês leitores. Após a leitura dsta matéria você, certamente, não estará apto a criar o super-hiper-big-game, a não ser que já o pudesse fazer antes, mas, ao menos, já terá se interessado pelo tema, e isto para mim será uma grande vitória.

Minha proposta é explicar algumas técnicas de criação de jogos, usando como base o nosso versátil MSX, e mais um jogo-exemplo, que será explicado no decorrer do texto. Não é um curso de criação de jogos. Toda a revista não seria suficiente para isto. Tentarei passar alguns conceitos e regras que poderão facilitar seu trabalho, principalmente se for novato no assunto. Se você já é um expert em games, faça como eu, divida com os outros seu conhecimento. CPU está aqui para isto.

Os jogos podem ser classificados em diversos tipos: de ação, de raciocínio, adventures, simuladores, etc. Os melhores, geralmente, contém todas estas características. Nesta matéria vou enfocar apenas os jogos de ação, que são os mais populares pela simplicidade das regras e por normalmente objetivarem apenas o lazer.

Não existem regras ou fórmulas fixas para se fazer um jogo, elas existem para fazer com que ele seja melhor. Entre as principais estão, a estética, a originalidade e a motivação.

Estética significa uma boa distribuição dos personagens, bons gráficos e boa visualização do que está ocorrendo no vídeo. A estética deve valer para o jogo todo e não só para a apresentação, como ocorre em muitos casos.

A motivação refere-se ao assunto tratado, como as batalhas espaciais intermináveis, por exemplo, que são a coqueluche dos fliperamas. Pessoalmente prefiro jogos mais calmos, tipo golfe, mas como cada um tem sua preferência, este aspecto deverá estar relacionado mais com o público a quem irá se dirigir a criação que com a preferência do criador.

Quanto à originalidade, não é quesito de escola de samba e sim a sua capacidade de criar ao invés de copiar. Não digo que você não deva tentar fazer seu PAC-MAN, apenas faça-o diferente do que todos nós conhecemos. Atualmente é difícil inventar algo que já não tenha sido feito pelos profissionais lá de fora, mas isto não nos impede de botar a cuca pra funcionar.

As características descritas acima não garantem um bom jogo, apenas fazem parte dele. Para se chegar ao produto final é necessário muito mais, tal como boas ferramentas de programação, e aí reside nosso maior problema: elas praticamente não existem no Brasil. Um simples editor de Sprites, que funcione decentemente, é raro de se encontrar. Isto nos obriga a criar nossas próprias ferramentas, e o que é pior, devido a anarquia e falta de respei-

to pelo trabalho alheio, as poucas ferramentas de boa qualidade, que facilitariam a criação de games, são guardadas a sete chaves pelos autores, que sentem um medo terrível de ver seus programas no mercado pirata.

Quanto a este pavor que alguns sentem em expor programas e idéias nada posso dizer. É resultado direto de uma má formação nesta área, que visa o individualismo, onde só o que pode vencer é o trabalho em equipe. Na criação de jogos a equipe é essencial, pois estão envolvidas características difíceis de serem encontradas em uma só pessoa.

Essa deficiência fica clara no caso de jogos que exigem gráficos de qualidade, que nem todos temos capacidade de criar nem mesmo com o melhor dos editores gráficos. Muitas vezes uma boa idéia é desperdiçada em programas com gráficos pobres, desprovidos de vida, geralmente por terem sido gerados por procressos não artísticos, tais como digitalizadores de imagem que criam figuras estereotipadas e sem graça. Um bom exemplo disto está nos jogos Goonies e Back to The Future do MSX. Uma simples olhada nas versões destes jogos para o ZX SPECTRUM (nosso TK90/95) mostra como um mesmo programa pode ser bem ou mal feito. As versões para MSX ficam ridículas e o MSX tem uma capacidade gráfica muito superior à do ZX SPECTRUM.

No Brasil ainda não se formaram equipes produtivas, voltadas para a área de jogos. Uma das poucas tentativas, efetuada pela conhecida DISPROSOFT, morreu junto com a empresa. As poucas equipes existentes voltam-se para a área editorial, de retorno mais garantido, ou ficam no nível do hobbysmo, sem divulgação e interesses comerciais. Isto não quer dizer que não existam profissionais neste setor, pelo contrário, quer dizer que eles existem e apenas não são prestigiados.

Estou me deixando levar pelo tema e fugindo um pouco de nosso objetivo inicial, que é mostrar como se faz um jogo, mas ficaria feliz em discutir as teorias anteriores como quem se interessar pelo assunto, escreva para CPU, que a gente continua o papo.

As técnicas que irei apresentar são baseadas em minha própria experiência e, naturalmente, não vem a ser nenhuma bíblia de programação de jogos. Você poderá usá-las de acordo com seus próprios métodos de trabalho, adaptando-as da forma mais conveniente. Todos os exemplos vão girar em torno de um mesmo programa que apresentarei a seguir.

Cavernas de Marte foi um dos jogos que cheguei a comercializar profissionalmente, ainda na época do ZX 81, os famosos TK's 81/82/83, irmãos mais novos dos TK's 90/95, que a Microdigital lamentavelmente deixou morrer, e digo lamentavelmente porque estas máquinas estão ainda em plena ascensão em alguns países da Europa, competindo até mesmo com PC e MSX, provavelmente ganhando deste último.

JUnto com CM (vamos chamá-lo assim para simplificar) fiz mais 3 jogos, PINBALL, VALKIRIE (baseado em um programa do TRS80/CP500, de Leo Christopherson, um guru dos jogos animados) e uma versão do Q*BERT do Atari, todos me deixaram bastante satisfeitos com seu resultado final, mas o CM é o meu preferido pois é o único que foi criado a partir do nada, além de ter sido meu primeiro jogo animado.

O ENREDO

CM foi desenvolvido inicialmente em BASIC e se tornou viável com um compilador. Depois foi criada uma nova versão em assembler. A história é simples. Um astronauta faz um pouso forçado em Marte e tem que reabastecer sua nave com urânio, que é facilmente encontrado por lá (isto na minha imaginação). Para pegar o urânio, ele deverá escavar o solo usando uma roupa apropriada. O único problema é que, além do urânio, existem também por lá algumas baratas super-desenvolvidas e que fazem seu lar justo nas cavernas onde tem o urânio. É claro que as baratas tem como prato principal astronautas escavadores. A única arma contra elas são pedras gigantes, mas estas pedras são uma faca de dois gumes, pois podem liquidar também com o astronauta. Como se já não bastassem tantos problemas, o coitado ainda tem uma escassa reserva de oxigênio, que acaba rapidinho. E tem mais. Ao morrer, a barata emite um chamado e logo aparece outra.

A estórinha aí de cima contém um dos principais ingredientes de uma receita de gama, o enredo. Por mais imbecil que um game possa parecer, ele precisa de um enredo para justificá-lo, mesmo que o enredo também seja imbecil. Só para exemplificar, quando o criador do SPACE INVADERS procurou alguém para bancar a comercialização do jogo nos Estados Unidos, encontrou diversas portas fechadas, devido principalmente a pouca credibilidade que poderia ter um jogo com alienígenas que nunca acabam e uma nave atirando indefinidamente. Pois bem, ele sozinho bancou o jogo e, com o que ganhou, criou a ATARI, depois vendida à Warner Brothers.

Criar um enredo é fácil. Pense no que gostaria que acontecesse no seu jogo e coloque no papel. Depois, vá apagando todas as coisas impossíveis de fazer. Se não sobrar nada, volte ao início do parágrafo.

O algoritmo acima não é piada. Experimente o processo e verá que fazer um jogo é apenas questão de permitir que o mesmo tenha condições de ser acabado. Criações muito complexas e cheias de sofisticações tendem a permanecer indefinidamente inacabadas.

A ANIMAÇÃO

Sem dúvida, o ponto alto de um jogo animado é a movimentação e desenhos dos personagens, mas isto não implica em quadros desenhados pela equipe Disney e animações produzidas por Steven Spielberg – ambos já tentaram produzir games e o resultado não foi dos melhores – basta uma boa dose de auto-sugestão que deverá ser dirigida ao usuário do game. Se você conseguir convencê-lo de que um asterisco é uma barata, a letra "A" é um astronauta e a letra "O" é uma pedra, ele irá ver estes personagens na tela. Fiz isto no CM, porque o ZX 81 não tinha resolução gráfica alguma e isto justificou tudo. Repare que o "A" parece um carinha com pernas e capacete, o "*" tem todas as

perninhas de uma barata e o "O" parece uma pedra, ficou convencido? Você está tentado a acreditar, mas este truque jamais iria funcionar para o MSX, isto porque existe a possibilidade de desenhar um astronauta e os outros personagens. Mas o que vale é a improvisação, a criatividade. Colocar a cara do Neil Armstrong em um shape não garante o sucesso de nenhum game.

Na versão MSX do CM, para fazer as pedras, a barata e o astronauta (e outros personagens da paisagem), optei pela redefinição de caracteres, que torna o jogo mais fácil de ser manipulado. Usei também os sprites para fazer uma navezinha espacial, bem simples, cuja definição está nas linhas DATA da listagem em BASIC. Se você domina a técnica de criação de sprites, poderá modificá-la à vontade, sem que isto interfira no andamento do jogo.

Repare que o astronauta e a barata se movimentam ao caminhar. Isto é facilmente conseguido pela troca da matriz dos respectivos caracteres que representam cada um. Esta é uma das técnicas mais rudimentares de animação, ou seja, imprimir várias seqüências de figuras levemente alteradas, para dar a sensação de movimento. O mais complicado é controlar a movimentação destas figuras, e aí entraria uma ferramenta de animação, mas isto é assunto para outra matéria. Para entender o funcionamento da animação, estude as linhas da rotina em LM (linguagem de máquina), da listagem 3, que estão comentadas de forma a facilitar seu entendimento.

A VELOCIDADE

A velocidade é muito importante em um jogo animado. No processamento deverá ser a maior possível, sendo que para se conseguir efeitos que exijam menor velocidade devem ser usados loops de espera. O BASIC (mesmo o do MSX) nos impõe algumas restrições e, para contorná-las, podemos usar rotinas em assembler ou compiladores que permitem uma maior velocidade de processamento, o CM usa estas duas técnicas, foi desenvolvido de forma a ser compatível com o TURBO BASIC, comercializado pela PAULISOFT, e usa rotinas auxiliares em assembler.

O aspecto da velocidade não está ligado apenas às características da máquina ou soft utilizado. É importante que sejam observadas as regras da física no vídeo. Já vi inúmeros jogos profissionais com intervalos de tempo durante as movimentações. Tais intervalos são gerados apenas por falta de planejamento nas ações que estão ocorrendo no vídeo. Imagine uma batalha espacial, onde a sua nave fique parada enquanto a nave inimiga atingida explode, ou um avião que pare no ar enquanto seu míssiil está descendo. Sei que um SEA HARRIER inglês, ou um helicóptero podem "parar" no ar, mas isto nunca deve acontecer sem planejamento. Todos os movimentos dos diversos personagens de um bom jogo devem ser independentes entre si.

Observe no CM que a movimentação das pedras é feita pela rotina auxiliar em LM, para que não interfira no andamento do jogo.

DIFICULDADE

O que prende um jogador a um jogo é, normalmente, o grau de dificuldade que o mesmo apresenta, mas aí entra um paradoxo. Se a dificuldade for muita, o jogador desiste. A sutileza de um jogo está em atingir o ponto máximo de dificuldade tolerada pelo usuário. Um bom exemplo disto é o SPACE INVADERS, que não tem outro atrativo que não seja a impossibilidade de vencê-lo e o aumento gradativo da sua dificuldade, até um ponto insuportável, até para o mais viciado dos jogadores. A dificuldade do CM é semelhante. Inicialmente, a barata é lenta até onde não cause tédio ao jogador, mas, no decorrer do jogo, vai alcançando uma velocidade de fazer inveja ao Ben Johnson (com anabolizante e tudo). No TK isto era facilmente obtido com o assembler, mas no MSX tais recordes de velocidade só poderão ser atingidos com o uso do TURBO, no BASIC normal provavelmente o jogador vai fazer a barata de boba.

Outro lado importante da dificuldade de vencer um jogo, e que afeta sensivelmente sua aceitação, é a morte do jogador. Jogos que obrigam o jogador a voltar ao início da partida a cada tiro levado ou nave destruída, normalmente não são bem aceitos. Um estudo dos pontos de retorno vai influir decisivamente na qualidade do seu programa. O ideal é ter pontos de reinício no meio do trajeto, que uma vez atingidos constituam uma fase completada. Um bom exemplo disto é o jogo Nemesis, que volta ao início de cada bloco atingido após a morte do jogador, mas não volta para o ponto inicial do jogo.

O CENÁRIO

A complementação de qualquer jogo é o cenário. Deve ser despojado o suficiente para não confundir o jogador – se você já jogou SKY JAGUAR, sabe do que estou falando – e ao mesmo tempo situá-lo o melhor possível no ambiente do jogo. Neste item, a veia artística de cada um irá falar mais alto e também os recursos gráficos de seu equipamento. O MSX não facilita a criação de cenários mas, com imaginação, qualquer problema pode ser solucionado, pois aqui também vale mais a originalidade que a sofisticação.

CM não tem um cenário dos mais ricos, mas isto é justificado pelo próprio enredo do jogo, pois o ambiente se passa no solo do planeta Marte. Mesmo assim, uma pequena nave na superfície ajuda a compor e valorizar o ambiente, que é dos mais fáceis de ser criado. Mantive nesta versão a mesma representação do solo usado na versão original. Sinta-se a vontade para modificá-lo, pois são gerados pelos caracteres normais do micro que podem ser redefinidos. O caracter é o do código 204 e não deve ser substituído por outro, pois é checado pelas rotinas do BASIC e LM.

Outra característica importante de um cenário é a sua portabilidade, ou seja, a facilidade de se compor várias telas diferentes com umas poucas figuras. Alguns jogos, como o TITANIC, cujo mapa está nas páginas 49 e 50 de CPU nº 8, possuem inúmeras telas diferentes, mas montadas sempre com as mesmas figuras. Tais figuras devem ser planejadas de forma a

poder serem repetidas várias vezes, até em uma mesma tela, compondo um ambiente que não ocupe muito espaço de armazenamento, pois deve ser levado em conta os limites de memória RAM e de disco que irão ser ocupados pelo programa. Jogos muito grandes não são necessariamente sinônimos de bons jogos.

No CM, o problema das paisagens foi resolvido com um simples algoritmo de montagem randômica para cada fase do jogo. Veja na listagem BASIC a parte intitulada MONTA A PAISAGEM.

A INTERAÇÃO DOS PERSONAGENS

Um bom jogo, deve testar exaustivamente todas as opções de movimentação possíveis entre os personagens, tratando cada qual de acordo com suas características. Verificações mal planejadas podem tornar um jogo interessante em um fracasso. Imagine um tiro que passe atraves do inimigo sem afetá-lo. Isto iria tirar o prazer de qualquer partida.

A posição de todos os personagens em movimento devem ser verificadas constantemente, tiros, o personagem que representa o jogador, os "inimigos", enfim tudo que se mover no vídeo, e até alguns objetos fixos, tem que ser verificados e tratados de acordo com as convenções do jogo. Isto representa a parte mais completa da criação e, normalmente, é o ponto final dos desavisados, pois se a interação não for obtida, o jogo não fica pronto.

A maneira mais prática de se testar cada personagem é criar um mapa de coordenadas. No MSX, dependendo do SCREEN escolhido para o jogo, este mapa pode ser a própria VDP, o CM foi estruturado para usar a SCREEN 1, e o teste dos personagens é feito simplesmente lendo o VDP. Optei por esta forma para poder ler facilmente, via coordenadas, o conteúdo de cada posição no vídeo.

A rotina de verificação da queda das pedras, foi desenvolvida em assembler, pois é solicitada constantemente e influiria na velocidade do jogo se fosse desenvolvida de outra forma. Seu funcionamento obedece um algoritmo simples. O vídeo é varrido de baixo para cima e sempre que é encontrado o caracter "@" (código 64), que representa as pedras, é verificado o caracter abaixo da mesma, para ver se ela deve ou não cair. Caso a barata seja atingida, é devolvido o código 1 na posição zero da VDP (esta posição foi escolhida ao acaso, poderia ser qualquer outro endereço).

No MSX, os personagens poderiam ser verificados usando os comandos de

```
                 VARIAVEIS E POSIÇÕES IMPORTANTES

U(5) = Situação das baterias, se = 32 está carregada.
U    = Indica a bateria da vez.
O    = Nível do oxigênio. (Descubra o que mais)
K    = última tecla cursora pressionada.
PA   = Posição do astronauta na VRAM.
MA   = Memória da posição do astronauta.
PB   = Posição do morcego na VRAM.
MA   = Memória da posição do morcego.
LA   = Coordenada de linha do astronauta.
LB   = Coordenada de linha do morcego.
CA   = Coordenada de coluna do astronauta.
CB   = Coordenada de coluna do morcego.
```

Os endereços 9 e 10 da VRAM, são utilizados para trocar parâmetros com a rotina em assembler. O endereço 9, serve para passar a direção de movimento do astronauta. Na saída da rotina assembler se o VPEEK(0)=1 é significa que o morcego foi atingido por uma pedra e se VPEEK(10)=1 quer dizer que há uma pedra sobre o astronauta, caso ele não se movimente poderá morrer.

No texto do artigo, o jogo se refere a uma "aranha gigante" e realmente no jogo original era uma aranha. Resolvi trocar a personagem por um morcego em homenagem aos 50 anos de BATMAN que é a capa deste número.

Caso deseje maiores informações sobre criação de jogos escreva para o autor, use o endereço da CPU.

colisão de sprites, desde que estes fossem usados. Mas, usando os caracteres, foi mantida a portabilidade do programa com a sua versão original.

Um bom teste de movimentação, volto a repetir, é imprescindível para se fazer um bom jogo. A lógica utilizada é simples, mas sua aplicação pode adquirir proporções assustadoras, principalmente em jogos com muitos personagens. Não existe uma fórmula para se fazer verificações. Cada programador usa um método próprio e procura adaptá-lo a cada programa. O melhor professor é a tentativa e erro. Portanto, mãos-a-obra.

Estude a integração das rotinas basic/assembler do CM, e tente tirar daí seu próprio método, já que a fórmula utilizada neste jogo vale apenas para ele próprio.

CM não tem nenhum brinde, nenhuma recompensa. tinha no jogo comercial para ZX81 uma animação com a decolagem do astronauta partindo do perigoso planeta. Como esta versão é apenas um exercício para ilustrar uma matéria, deixei para você leitor a criação de um Gran-finale para o CM. Faça e envie para CPU. Sua rotina pode ir para a CBBS de CPU e você concorre a assinaturas da revista e disquetes com programas.

Ao chegar ao fim desta matéria, verifiquei que não deu nem para raspar de leve o universo da criação de jogos. Ainda há muito a dizer, muito a fazer e o espaço é pouco. Mas o assunto não irá morrer aqui. A partir do próximo número de CPU, haverá uma seção fixa, dedicada a criação de jogos. Você deve participar com a sua dúvida e com a sua solução.

Você poderá participar desta nova seão, enviando suas experiências na área de criação de jogos. Uma equipe de programadores irá auxiliá-lo a resolver os problemas encontrados e as soluções serão publicadas, tornando acessível a todos os conhecimentos deste universo tão fascinante. Participe desde já, enviando um nome para batizar esta nova seção de CPU e suas sugestões.

Para digitar o CAVERNAS DE MARTE, entre com os dados em assembler, usando um DEBUG ou compilador assembler. A listagem 2 contém os códigos em hexadecimal da rotina em LM e a listagem 3 contém os mesmos códigos em mnemônicos Z80. Use o que compreender melhor. O programa em BASIC da listagem 1, só funcionará em conjunto com a rotina em LM, mas você poderá testá-lo retirando a linha que contém a chamada A=USR(0), que está titulada como "CHAMADA DA ROTINA LM". Neste caso, deverá também retirar a primeira linha que carrega o bloco assembler. A chamada da primeira linha, pressupõe que você esteja usando disk-driver, caso não disponha deste periférico, acrescente o parâmetro "CAS:" ao nome do programa.

A listagem 4, contém uma descrição das variáveis usadas pelo BASIC, para que você possa mexer no programa. Terei prazer em responder as dúvidas dos leitores que se interessarem pelo assunto da criação de jogos e estarei presente na nova seção de CPU.

Infelizmente chegamos ao final, que, como já afirmei anteriormente, apenas arranhou o assunto. Espero ter ao menos conseguido despertar seu interesse, com o pouco que pude passar nestas linhas. Esta matéria se encerra aqui, mas o assunto não, e tenho certeza que vocês leitores, e eu, não vamos deixar esta peteca cair!

```
10 COLOR 3,1,1:SCREEN 1,3:WIDTH3
2:KEYOFF
20 CLEAR 500,&HC300:BLOAD"CM.COM
":DEFUSR=&HC300
30 RESTORE 2340
40 FOR F=9TO88:READ A$:VPOKE F,V
AL("&B"+A$):NEXT F:A$=""
50 A=USR(0)
60 ' ---------------------------
    DESABILITA CURSOR
   ---------------------------
70 POKE &HFCA9,0
80 ' ---------------------------
    REINICIO DO PROGRAMA
   ---------------------------
90 CLS
100 '---------------------------
    DEFINE VARIAVEIS
   ---------------------------
110 DIM U(5)
120 AA=32:U=1:LO=70
130 O=100
140 '---------------------------
    VERIFICA CARGA DAS BATERIAS
   ---------------------------
150 FOR F=1TO5:IF U(F)<>32THEN 1
80
160 NEXT F
170 GOTO 1910
180 '---------------------------
    MONTA A PAISAGEM
   ---------------------------
190 FOR F=6208TO6240:VPOKE F,0:N
EXT F:VPOKE 6271,0:VPOKE 6272,0
200 FOR F=4TO21
210 LOCATE 0,F:PRINT" ▌▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▐ "
220 NEXT F
230 LOCATE 0,22:PRINT" ──────────
─────────────────────";
240 '---------------------------
    TORNA RND MAIS ALEATóRIO
   ---------------------------
250 R=RND(-TIME)
260 FOR F=1TORND(1)*40+20
270 X=INT(RND(1)*28)+2
280 Y=INT(RND(1)*17)+4
290 LOCATE X,Y:PRINT"@";
300 NEXT F
310 '---------------------------
    MONTA O NINHO DO MORCEGO
   ---------------------------
320 X=INT(RND(1)*23)+2
330 Y=INT(RND(1)*9)+7
340 FOR F= Y TO Y+4
350 LOCATE X,F:PRINT"     ";
360 IF F=Y+2 THEN LOCATE X+2,F:P
RINT"#";
370 NEXT F
380 LOCATE X,Y:PRINT"*";
390 VPOKE 6144,0:VPOKE 0,0:VPOKE
 9,0
400 ----------------------------
    POSIçãO DO MORCEGO NA VRAM
   ---------------------------
410 PB=Y*32+X+6144
420 '---------------------------
    POUSO DA NAVE
   ---------------------------
430 PN=INT(RND(1)*20)+5
440 PN=PN*8
450 GOSUB 1830
460 PA=6242+PN/8:MA=PA:VPOKE PA,
38
470 SP=PA:PS=0
480 FOR F=1TO5:IF U(F)<>32 THEN
U(F)=F+48
490 NEXT F
500 '---------------------------
    ATUALIZAçãO DO PAINEL
   ---------------------------
510 O=O-.5:IF O<0 THEN GOTO 2010
520 U=U+1:IF U=6THEN U=1
530 VPOKE 6902+U,48+U
540 LOCATE 3,23:PRINT"OXIGêNIO";
INT(O);" ";
550 IF U(U)>32 THEN VPOKE 6902+U
,32
560 '---------------------------
    MOVIMENTAçãO DO ASTRONAUTA
   ---------------------------
570 K=STICK(0):IF K=0THEN670
580 IF K=1 THEN MA=PA-32
590 IF K=3 THEN MA=PA+1
600 IF K=5 THEN MA=PA+32
610 IF K=7 THEN MA=PA-1
620 IF PA<6272 AND K=1 THEN K=0
630 Z=VPEEK(MA):IF Z=42 THEN 223
0
640 IF Z=35 THEN U(U)=32:VPOKE 6
902+U,48+U:PS=SP
650 IF Z=32 OR Z=204 OR Z=35 THE
N VPOKE MA,38:VPOKE PA,32:PA=MA
660 IF PA=PS THEN 1870
670 VPOKE 9,K
680 A=USR(0)
690 IF VPEEK(9)=1 THEN VPOKE PB,
64:PB=6270:AA=-1:VPOKE 9,0
700 IF VPEEK(10)=1 AND K=5 THEN
GOTO 2140
710 IF VPEEK(10)=1 THEN GOTO 211
0 ELSE PP=0
720 '---------------------------
    TESTA MOVIMENTO DO MORCEGO
   ---------------------------
730 LA=INT((PA-6144)/32):CA=PA-6
144-LA*32
740 LB=INT((PB-6144)/32):CB=PB-6
144-LB*32
750 AM=AA
760 IF LA<>LB THEN 800
770 IF CA>CB THEN AA=1
780 IF CA<CB THEN AA=-1
790 GOTO 830
800 IF CA<>CB THEN 860
810 IF LA>LB THEN AA=32
820 IF LA<LB THEN AA=-32
830 Z=VPEEK(PB+AA):IF Z=38 THEN
2230
840 IF Z=32 THEN 1330
850 IF O<LO AND Z=204 THEN 1330
860 AA=AM
870 IF AA=32 THEN 980
880 IF AA=-32 THEN 1060
890 IF AA=1 THEN 1150
900 '---------------------------
    MORGEGO SEGUE A ESQUERDA
   ---------------------------
910 MD=32:IF LA<LB THEN MD=-32
920 Z=VPEEK(PB+AA):IF Z=38 THEN
2230
930 IF Z=32 THEN 1330
940 IF O<LO AND Z=204 THEN 1330
950 IF AA=(MD*-1)THEN AA=1:GOTO
1230
960 IF AA=-1THENAA=MD:GOTO920
970 AA=AA*-1:GOTO 920
980 '---------------------------
    MORCEGO SEGUE PARA BAIXO
   ---------------------------
990 MD=1:IF CA<CB THEN MD=-1
1000 Z=VPEEK(PB+AA):IF Z=38 THEN
 2230
1010 IF Z=32 THEN 1330
1020 IF O<LO AND Z=204 THEN 1330
1030 IF AA=(MD*-1) THEN AA=-32:G
OTO1230
1040 IF AA=32 THEN AA=MD:GOTO100
0
1050 AA=AA*-1:GOTO 1000
1060 '--------------------------
    MORCEGO SEGUE PARA CIMA
   --------------------------
1070 MD=1:IF CA<CB THEN MD=-1
1080 Z=VPEEK(PB+AA):IF Z=38 THEN
 2230
1090 IF Z=32 THEN 1330
1100 IF O<LO AND Z=204 THEN 1330
1110 IF AA=(MD*-1)THEN AA=32:GOT
O1230
1120 IF AA=-32 THEN AA=MD:GOTO 1
080
1130 AA=AA*-1:GOTO 1080
1140 MD=32:IF LA<LB THEN MD=-32
1150 '--------------------------
    MORCEGO SEGUE A DIREITA
   --------------------------
1160 MD=32:IF LA<LB THEN MD=-32
1170 Z=VPEEK(PB+AA):IF Z=38 THEN
 2230
1180 IF Z=32 THEN 1330
1190 IF O<LO AND Z=204 THEN 1330
1200 IF AA=(MD*-1)THEN AA=-1:GOT
O1230
1210 IF AA=1THEN AA=MD:GOTO1170
1220 AA=AA*-1:GOTO 1170
1230 AM=AA
1240 IF LA>LB THEN AA=32
1250 IF LA<LB THEN AA=-32
1260 IF CA>CB THEN AA=1
1270 IF CA<CB THEN AA=-1
1280 Z=VPEEK(PB+AA):IF Z=38 THEN
 2230
1290 IF Z=32 THEN 1330
1300 IF O<LO AND Z=204 THEN 1330
1310 AA=AM
1320 GOTO 500
1330 '--------------------------
    MOVE O MORCEGO
   --------------------------
1340 VPOKE(PB+AA),42:VPOKE PB,32
1350 PB=PB+AA:GOTO 500
1360 '--------------------------
    DEFINIçãO DOS SPRITES
   --------------------------
1370 DATA 0010000100000100
1380 DATA 0100000100000010
1390 DATA 0100101110010010
1400 DATA 1101011111101011
1410 DATA 1111111111111111
1420 DATA 1101100000011011
1430 DATA 1101100000011011
1440 DATA 1111110000111111
1450 DATA 0101111111111010
1460 DATA 0100011111100010
1470 DATA 0010001111000100
1480 DATA 0000000000000000
1490 DATA 0000000000000000
1500 DATA 0000000000000000
1510 DATA 0000000000000000
1520 DATA 0000000000000000
1530 DATA 0001000100001000
1540 DATA 0011000100001100
1550 DATA 0111101110011110
1560 DATA 1111011111101111
1570 DATA 1111111111111111
1580 DATA 1001100000011001
1590 DATA 0001100000011000
1600 DATA 0011110000111100
1610 DATA 0001111111111000
1620 DATA 0010011111100100
1630 DATA 0100001111000010
1640 DATA 0100000000000010
1650 DATA 1000000000000001
1660 DATA 1000000000000001
1670 DATA 0100000000000010
1680 DATA 1110000000000111
1690 '--------------------------
    GERA SPRITE DA NAVE 1
   --------------------------
1700 FOR F=1TO16
1710 READ D$
1720 A$=A$+CHR$(VAL("&B"+LEFT$(D
$,8)))
1730 B$=B$+CHR$(VAL("&B"+RIGHT$(
D$,8)))
1740 NEXT F
1750 SPRITE$(0)=A$+B$:RETURN
1760 '--------------------------
    GERA SPRITE DA NAVE 2
   --------------------------
1770 FOR F=1TO16
1780 READ D$
1790 A$=A$+CHR$(VAL("&B"+LEFT$(D
$,8)))
1800 B$=B$+CHR$(VAL("&B"+RIGHT$(
D$,8)))
1810 NEXT F
1820 SPRITE$(0)=A$+B$:RETURN
1830 RESTORE 1370
1840 GOSUB 1690:FOR F=-23 TO 0:P
UTSPRITE0,(PN,F):FOR G=1TO40:NEX
T G:A=USR(0):NEXT F
1850 A$="":B$="":GOSUB 1760
1860 A$="":B$="":RETURN
1870 '--------------------------
    ASTRONAUTA VENCE UMA FASE
   --------------------------
1880 VPOKE PA,32:RESTORE 1360
1890 GOSUB 1690:FOR F=0TO -23 ST
EP -1:PUTSPRITE0,(PN,F):FOR G=1T
O40:NEXT G:NEXT F
1900 A$="":B$="":VPOKE PB,32:GOT
O 130
1910 '--------------------------
    ASTRONAUTA VENCE O JOGO
   --------------------------
1920 CLS:SPRITE$(0)="":PRINT"  *
** VITóRIA DO ASTRONAUTA ***"
1930 PRINT:PRINT" NESTE PONTO VO
Cê PODE FAZER UMAANIMAçãO E DEPO
IS VOLTAR AO JOGO"
1940 LOCATE 0,16:PRINT"TECLE RET
URN PARA VOLTAR AO JOGO"
1950 PRINT"TECLE <L> PARA LISTAR
"
1960 C=0
1970 K$=INKEY$:IF K$=CHR$(13) TH
EN RUN 80
1980 C=C+1:IF C=15 THEN A=USR(0)
:C=0
1990 IF K$<>"L" AND K$<>"l" THEN
 1970
2000 CLS:LIST 1910-1990
2010 '--------------------------
    ACABA O OXIGêNIO
   --------------------------
2020 CLS:SPRITE$(0)="":PRINT"
***  ACABOU O OXIGêNIO    ***"
2030 PRINT:PRINT" NESTE PONTO VO
Cê PODE FAZER UMAANIMAçãO E DEPO
IS VOLTAR AO JOGO"
2040 LOCATE 0,16:PRINT"TECLE RET
URN PARA VOLTAR AO JOGO"
2050 PRINT"TECLE <L> PARA LISTAR
"
2060 C=0
2070 K$=INKEY$:IF K$=CHR$(13) TH
EN RUN 80
2080 C=C+1:IF C=15 THEN A=USR(0)
:C=0
2090 IF K$<>"L" AND K$<>"l" THEN
 2070
2100 CLS:LIST 2010-2100
2110 '--------------------------
    PEDRA PEGA ASTRONAUTA
   --------------------------
2120 PP=PP+1:IF PP>1 THEN 2130 E
LSE 720
2130 IF K>0 THEN 720
2140 CLS:SPRITE$(0)="":PRINT"
*** UMA PEDRA TE PEGOU    ***"
2150 PRINT:PRINT" NESTE PONTO VO
Cê PODE FAZER UMAANIMAçãO E DEPO
IS VOLTAR AO JOGO"
2160 LOCATE 0,16:PRINT"TECLE RET
URN PARA VOLTAR AO JOGO"
2170 PRINT"TECLE <L> PARA LISTAR
"
2180 C=0
2190 K$=INKEY$:IF K$=CHR$(13) TH
EN RUN 80
2200 C=C+1:IF C=15 THEN A=USR(0)
:C=0
2210 IF K$<>"L" AND K$<>"l" THEN
 2190
2220 CLS:LIST 2110-2220
2230 '--------------------------
    MORCEGO PEGA ASTRONAUTA
   --------------------------
2240 CLS:SPRITE$(0)="":PRINT"
*** O MORCEGO TE PEGOU    ***"
2250 PRINT:PRINT" NESTE PONTO VO
Cê PODE FAZER UMAANIMAçãO E DEPO
IS VOLTAR AO JOGO"
2260 LOCATE 0,16:PRINT"TECLE RET
URN PARA VOLTAR AO JOGO"
2270 PRINT"TECLE <L> PARA LISTAR
"
2280 C=0
2290 K$=INKEY$:IF K$=CHR$(13) TH
EN RUN 80
2300 C=C+1:IF C=15 THEN A=USR(0)
:C=0
2310 IF K$<>"L" AND K$<>"l" THEN
 2290
2320 CLS:LIST 2230-2320
2330 '--------------------------
    DEFINIçãO DO URâNIO
   --------------------------
2340 DATA 00001000
2350 DATA 01000000
2360 DATA 00011001
2370 DATA 00111100
2380 DATA 00111100
2390 DATA 10011001
2400 DATA 00000000
2410 DATA 00010000
2420 DATA 00100000
2430 DATA 00000010
2440 DATA 00011000
2450 DATA 00111100
2460 DATA 10111101
2470 DATA 00011000
2480 DATA 00000000
2490 DATA 00100100
2500 '--------------------------
    DEFINIçãO DO MORCEGO
   --------------------------
2510 DATA 10000001
2520 DATA 10000001
2530 DATA 01000010
2540 DATA 01011010
2550 DATA 00111100
2560 DATA 00111100
2570 DATA 00011000
2580 DATA 00100100
2590 DATA 00000000
2600 DATA 00100100
2610 DATA 01111110
2620 DATA 01011010
2630 DATA 10011001
2640 DATA 10100101
2650 DATA 10000001
2660 DATA 01000010
2670 '--------------------------
    DEFINIçãO DA PEDRA
   --------------------------
2680 DATA 10001100
2690 DATA 01110010
2700 DATA 01000001
2710 DATA 10000001
2720 DATA 10000001
2730 DATA 01000010
2740 DATA 01000110
2750 DATA 00111101
2760 '--------------------------
    DEFINIçõES DO ASTRONAUTA
   --------------------------
2770 DATA 00011000
2780 DATA 00100100
2790 DATA 00100100
2800 DATA 01111110
2810 DATA 10111101
2820 DATA 00111100
2830 DATA 00100100
2840 DATA 01100110
2850 '--------------------------
    MOVIMENTOS BASICOS PASSO 1
   --------------------------
2860 DATA 00011000
2870 DATA 00100100
2880 DATA 00100100
2890 DATA 00111100
2900 DATA 00111100
2910 DATA 00111100
2920 DATA 00010000
2930 DATA 00011000
2940 '--------------------------
    MOVIMENTOS BASICOS PASSO 2
   --------------------------
2950 DATA 00011000
2960 DATA 00100100
2970 DATA 00100100
2980 DATA 01111100
2990 DATA 10111100
3000 DATA 00111100
3010 DATA 00011010
3020 DATA 00110100
3030 '--------------------------
    MOVIMENTOS BASICOS PASSO 3
   --------------------------
3040 DATA 00011000
3050 DATA 00100100
3060 DATA 00100100
3070 DATA 00111100
3080 DATA 00111100
3090 DATA 00111101
3100 DATA 01000010
3110 DATA 00100000
3120 '--------------------------
    MOVIMENTOS BASICOS PASSO 4
   --------------------------
3130 DATA 00011000
3140 DATA 00100100
3150 DATA 00100100
3160 DATA 01111100
3170 DATA 10111100
3180 DATA 00111100
3190 DATA 00011010
3200 DATA 00110100
```

LISTAGEM EM HEXADECIMAL

```
END.    BYTES                           SOMA
C300 - 3A 43 C5 B7 C2 58 C3 3C  = 0412
C308 - 32 43 C5 21 09 00 CD 34  = 0265
C310 - C5 06 50 21 49 C5 DB 98  = 03BD
C318 - 77 23 10 FA 3E 26 11 71  = 028A
C320 - C5 CD 10 C5 3E 2A 11 59  = 0339
C328 - C5 CD 10 C5 3E 23 11 49  = 0322
C330 - C5 CD 10 C5 3E 40 11 69  = 035F
C338 - C5 CD 10 C5 21 79 C5 11  = 03D7
C340 - 99 C5 CD 50 C4 21 79 C5  = 049E
C348 - 11 D9 C5 CD 8C C4 21 99  = 0486
C350 - C5 11 B9 C5 CD 8C C4 C9  = 053A
C358 - AF 32 45 C5 32 46 C5 21  = 0349
C360 - 09 00 CD 34 C5 DB 98 32  = 0374
C368 - 44 C5 CD F7 C3 CD 81 C3  = 05A1
C370 - 21 09 00 CD 29 C5 3A 45  = 0264
C378 - C5 D3 98 3A 46 C5 D3 98  = 04E0
C380 - C9 3A 48 C5 B7 28 09 3D  = 0335
C388 - 32 48 C5 11 59 C5 18 07  = 028D
C390 - 3C 32 48 C5 11 61 C5 3E  = 02F0
C398 - 2A CD 10 C5 3A 47 C5 B7  = 03C9
C3A0 - 28 09 3D 32 47 C5 11 49  = 0206
C3A8 - C5 18 07 3C 32 47 C5 11  = 026F
C3B0 - 51 C5 3E 23 CD 10 C5 3A  = 0353
C3B8 - 44 C5 FE 01 28 1A FE 03  = 034B
C3C0 - 28 25 FE 05 28 17 FE 07  = 0294
C3C8 - 28 18 3E 26 11 71 C5 CD  = 02B8
C3D0 - 10 C5 3E 18 32 3D C5 C9  = 0328
C3D8 - 11 B9 C5 18 0D 11 D9 C5  = 0363
C3E0 - 18 08 11 99 C5 18 03 11  = 01BB
C3E8 - 79 C5 CD 42 C4 26 00 6F  = 03A6
C3F0 - 19 EB 3E 26 C3 10 C5 21  = 0321
C3F8 - C0 1A CD 34 C5 DB 98 FE  = 0511
C400 - 40 CC 0B C4 FE 00 C8 2B  = 03CC
C408 - 18 F0 C9 11 20 00 19 CD  = 02E8
C410 - 34 C5 DB 98 FE 2A 28 0C  = 03C8
C418 - FE 26 CC 3C C4 FE 20 28  = 0436
C420 - 0A ED 52 C9 F5 3E 01 32  = 0378
C428 - 45 C5 F1 CD 29 C5 3E 40  = 0434
C430 - D3 98 ED 52 CD 29 C5 3E  = 04A3
C438 - 20 D3 98 C9 3E 01 32 46  = 030B
C440 - C5 C9 3A 3D C5 C6 08 FE  = 0496
C448 - 20 20 01 AF 32 3D C5 C9  = 02ED
C450 - 06 20 7E 0E 00 CB 47 28  = 01EC
C458 - 02 CB F9 CB 4F 28 02 CB  = 03D5
C460 - F1 CB 57 28 02 CB E9 CB  = 04BC
C468 - 5F 28 02 CB E1 CB 67 28  = 038F
C470 - 02 CB D9 CB 6F 28 02 CB  = 03D5
C478 - D1 CB 77 28 02 CB C9 CB  = 049C
C480 - 7F 28 02 CB C1 79 12 23  = 02E3
C488 - 13 10 C7 C9 3E 04 32 3E  = 0265
C490 - C5 22 3F C5 ED 53 41 C5  = 0431
C498 - 11 F9 C5 06 08 AF 12 13  = 02B1
C4A0 - 10 FC 06 08 11 F9 C5 EB  = 03D4
C4A8 - 0E 01 E5 1A CB 7F C4 EF  = 040B
C4B0 - C4 23 CB 77 C4 EF C4 23  = 04C3
C4B8 - CB 6F C4 EF C4 23 CB 67  = 0506
C4C0 - C4 EF C4 23 CB 5F C4 EF  = 0577
C4C8 - C4 23 CB 57 C4 EF C4 23  = 04A3
C4D0 - CB 4F C4 EF C4 23 CB 47  = 04C6
C4D8 - C4 EF C4 79 07 4F 13 E1  = 043A
C4E0 - 10 C8 CD F5 C4 3A 3E C5  = 049B
C4E8 - 3D C8 32 3E C5 18 A9 F5  = 03F0
C4F0 - 7E B1 77 F1 C9 2A 3F C5  = 048E
C4F8 - 01 08 00 09 22 3F C5 21  = 0159
C500 - F9 C5 ED 5B 41 C5 ED B0  = 05A9
C508 - ED 53 41 C5 2A 3F C5 C9  = 043D
C510 - 26 00 6F D5 16 00 5F 06  = 01E5
C518 - 07 19 10 FD D1 CD 29 C5  = 03B9
C520 - 06 08 1A D3 98 13 10 FA  = 02B0
C528 - C9 F3 7D D3 99 7C F6 40  = 0557
C530 - D3 99 FB C9 F3 7D D3 99  = 060C
C538 - 7C D3 99 FB C9 18 00 00  = 03C4
C540 - 00 00 00 00 00 00 00 00  = 0000
```

NAVY MOVES II

CARLOS DOS SANTOS

O objetivo deste artigo é de apenas complementar o que foi publicado na CPU nº 8, onde mostrava um mapa das telas apresentadas pelo jogo, alguns macetes e palavras "chave" a serem introduzidas nos terminais de computador espalhados pelas salas do "submarino", onde se desenrola o jogo.

No final deste artigo, apresentamos o mesmo mapa publicado na CPU nº 8 em tamanho reduzido, tendo como alteração as letras I – sala de início do jogo; F – sala de final do jogo; e B – sala onde a bomba tem que ser ativada.

O primeiro objetivo é conseguir as 3 senhas (números) para acessar os terminais do computador (no jogo). Conforme devidamente explicado na CPU nº 8, estas senhas estão com o 1º oficial e os de máquinas e transmissão, sendo que os dois oficiais de máquinas têm a mesma senha e o mesmo ocorre com os dois oficiais de transmissão. Isto quer dizer que, se você matar os dois, pegue a senha de um e do outro pegue a "vida" (maleta de primeiros socorros). Mas, se você conseguir matar o capitão, com a senha dele você pode saber a senha de todos os outros sem precisar matá-los. O segundo objetivo é alcançar o "terminal de computador" ao lado da sala "B" e, então, operar o terminal da seguinte forma:

1º) Se você tem a senha do capitão e falta alguma senha, digite "DERD". O terminal vai pedir a senha do capitão e, estando esta correta, vai informar 4 algarismos relativos às senhas do capitão, 1º oficial, oficial de máquinas e oficial de transmissão (nessa ordem).

2º) Tendo todas as 3 senhas dos oficiais, ponha o submarino na superfície e pare os motores. Para fazer isto, digite os comandos "EMER" e, depois, "PARA MOTO". Estes comandos vão exigir a senha do 1º oficial.

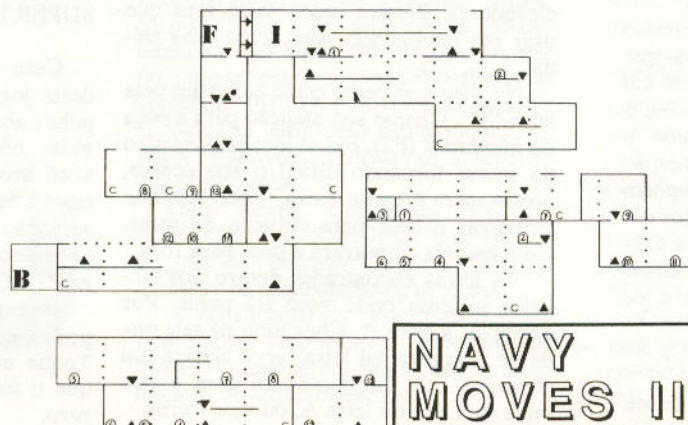
3º) Abra a porta da sala onde deverá explodir a bomba, digitando "ABRI PUER", o que exigirá a senha do oficial de máquinas.

4º) Encerre o "papo" com o "terminal do computador", digitando "FIM" e vá até o canto esquerdo da sala da bomba. Neste momento, a bomba é ativada e é mostrada uma mensagem na tela. Ao ser teclado qualquer coisa, iniciar-se-á a contagem regressiva de dois minutos para você sair fora deste submarino.

Porém, você ainda tem que avisar ao seu mini-submarino para vir buscá-lo e, para isto, novamente será necessário usar um "terminal de computador" qualquer.

5º) Chame o seu mini-submarino para resgatá-lo digitando no "terminal do computador" a palavra "TRAN", o que exigirá a senha do oficial de transmissão e, então, o computador ficará aguardando a mensagem que você tem que transmitir. Digite o seguinte: "OABERBYAMD" (que significa "bombardeia" com as letras misturadas) e depois encerre o "papo" com o computador, digitando "FIM". Um detalhe: o relógio da bomba fica parado enquanto você estiver utilizando o "terminal do computador".

6º) O negócio, agora, é correr. Não perca tempo, deitando para fugir das balas, fuzile todo o mundo que aparecer na frente e atrás e vá até a sala "F" e, lá chegando, dirija-se para o canto esquerdo e cruze os dedos. Se houver tempo hábil, o seu mini-submarino aparecerá e você será, automaticamente, transportado para dentro dele e será mostrada a explosão da bomba, saudando-o com efusivos cumprimentos e, posteriormente, você estará desembarcando numa ensolarada ilha da América Central, onde dançarinas nativas o saudarão.

Boa sorte.

BATMAN

Luis Fernando Fiacadori

Você estava em sua casa, curtindo uma tarde tranqüila. Descansando um pouco na sua identidade secreta de Bruce Wayne. De repente, você ouve o telefone tocar. Antes de atender o telefone, você já sabe que é mais encrenca para você resolver.

Não deu outra. Era o delegado de Gothan City, que quer que você e o Garoto Prodígio vão rápido para a chefatura de polícia. Sem questioná-lo, você diz que vai assim que puder e bate o telefone na cara do delegado. O que BATMAN naõ sabe é que a voz que ouviu ao telefone não era do delegado e sim do perverso PINGÜIM.

A primeira coisa a fazer é encontrar sua BATroupa no meio da tremenda bagunça que está sua casa, pois seu mordomo Alfred acaba de ir embora. Ele queria um aumento de salário, mas você, como é um Super-Herói, se negou a pagá-lo.

Depois de muita procura, você encontra um de seus uniformes; só que ele está um pouquinho sujo e amarrotado.

Você desce para a BATcaverna e percebe que seu BATmóvel não está no lugar que deveria estar. Repara que sumiram também com suas BATbotas, sua BATsacola, seu BATtubo e seu BATcinto de utilidades.

Você esperava encontrar o Garoto Prodígio aqui, mas no seu lugar você encontra um bilhete que diz o seguinte:

"Não foi fácil cavar um túnel do nosso QG até a BATcaverna mas, agora que conseguimos, temos um grande trunfo em nossas mãos. Pegamos seu BATmóvel, e raptamos Robin. Nós desmontamos o BATmóvel em sete (7) partes, e as espalhamos nas inúmeras salas que compõem o labirinto. Labirinto este que fizemos especialmente para você. Sendo que a carcaça do BATmóvel se encontra no nosso QG. Roubamos seus BATobjetos e os deixamos espalhados pelas salas."

De seus queridos vilões:

Ass. Pingüim e Coringa.

P.S. – Encontre logo as peças e monte seu BATmóvel rápido, pois só assim entregaremos Robin vivo.

DICAS

O Garoto Prodígio foi levado para o QG, juntamente com a carcaça do BATmóvel. Os vilões o amarraram e o jogaram no chão. Em seguida foram embora, para armar armadilhas para o BATMAN. O que você e os vilões não sabem é que o Garoto Prodígio escapou das cordas e fugiu. Como ele nunca encontraria BATMAN naquele imenso labirinto, quis ajudá-lo de qualquer forma. Então, espalhou diversos bonecos pelo labirinto. Esses bonecos se parecem com um MINI-BATMAN. Eles tem uma aparência inofensiva, mas lhe darão grandes poderes. Tais como:

– aumento da potência de seus pulos.
– imunidade total perante seus inimigos.
– aumento de velocidade. Com isso você andará mais rápido.

IMPORTANTE: Os poderes acima são por tempo determinado.

Existem salas indicadas por setas para cima (↑) e para baixo (↓), de acordo com o mapa, onde você poderá subir para um piso superior, ou mesmo cair para um andar inferior do labirinto. Nem sempre existem elevadores. Nestes casos você terá que usar sua inteligência para achar uma saída.

No mapa, as peças estão indicadas pela letra "P". Chamo sua atenção para a peça de número 2 (P2). Ela se localiza embaixo da mesa; tornando difícil o seu acesso, pois a mesa é muito baixa. Você terá que empurrar a bola para debaixo da mesa. Com isso ela empurrará a peça para fora.

As letras encontradas dentro dos círculos indicam onde você irá parar. Por exemplo, a letra A. Chegando na sala onde se encontra tal letra, você achará um elevador, que lhe conduzirá para a sala onde está a outra letra A, ou vice-versa.

Cada um dos quatro BATobjetos que sumiram da BATcaverna tem funções muito importantes, tais como:

– BATbotas – Sem elas você não poderá pular.

– BATsacola – Onde você poderá carregar bules, xícaras, banquinhos, etc. Eles são de extrema importância pois poderão lhe servir para alcançar um lugar alto, como por exemplo uma porta, ou mesmo um andar superior. Importante: Esses objetos não podem ser carregados dentro da BATsacola de uma sala para outra.

– BATtubo – Com esse tubo gravitacional, você controlará a direção de seus saltos.

– BATcinto de Utilidades – O mais importante de todos. Com ele seus saltos serão mais longos.

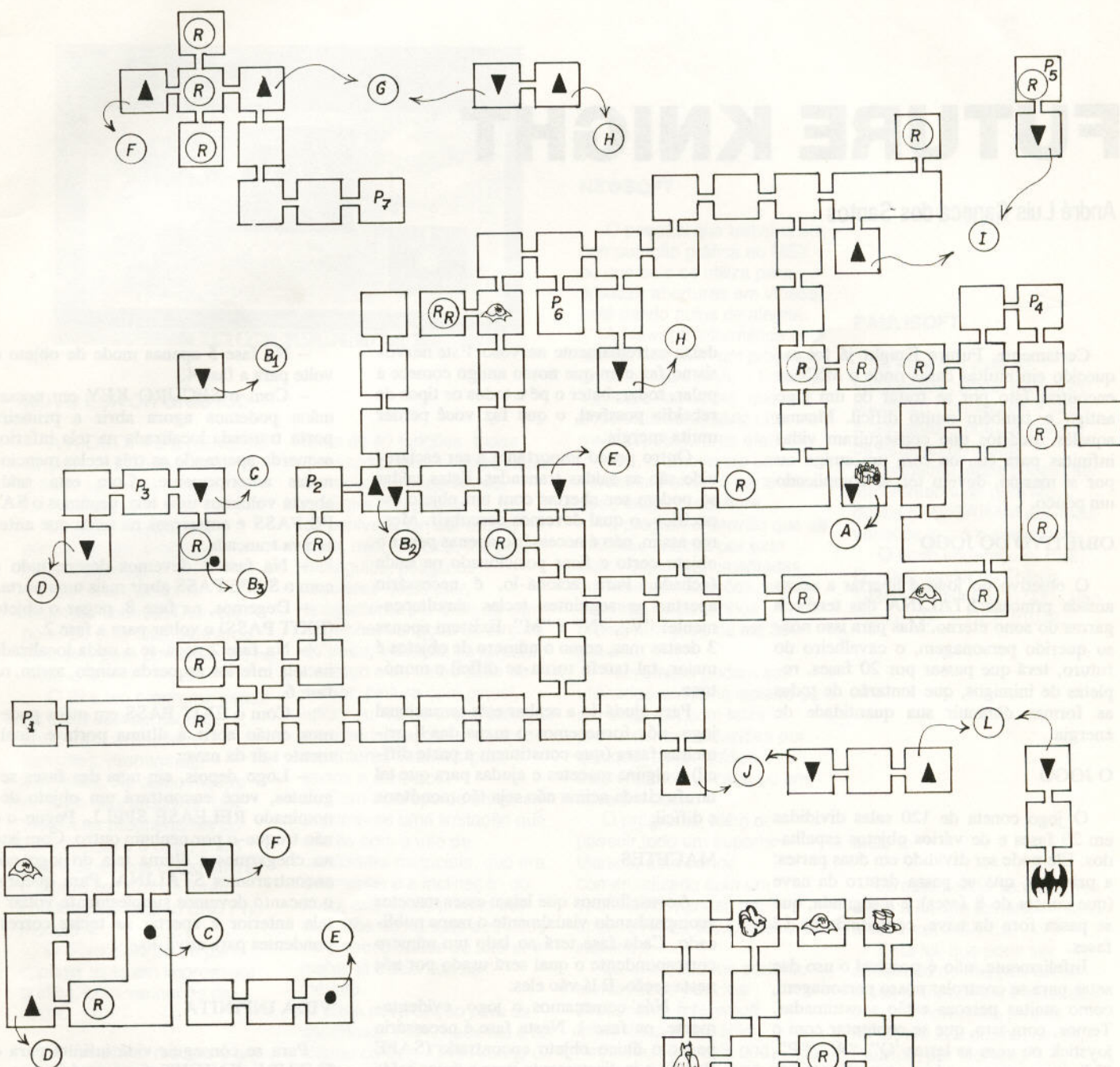
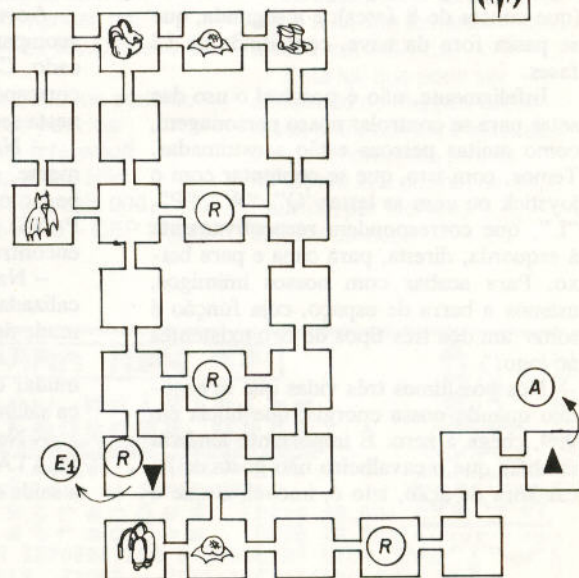
Não se esqueça de passar este jogo pressionando a tecla CONTROL do teclado quando ligar o micro.

FIM DO JOGO

Para se chegar no final do jogo, você deverá pegar os quatro (4) objetos que sumiram da BATcaverna, encontrar os vinte e cinco (25) MINI-BATMAN, achar as sete (7) peças do BATmóvel e montá-lo na sala, que de acordo com o mapa, está indicado pela letra "F".

SUPER DICA

Caso você ache difícil chegar no final deste jogo, aí vai uma grande ajuda. Os pokes abaixo permitem que o número de vidas não diminua, bem como deixará você invulnerável, podendo, assim, tocar em qualquer objeto ou local. A única observação a ser feita é que a invulnerabilidade deixa de protegê-lo quando ela atinge ZERO, retornando novamente a 99.

Isto se dee à possibilidade de que você pode cair em uma sala que não tem saída. Toque em um dos inimigos ou em algo que o matará quando a energia atingir a zero.

FUTURE KNIGHT

André Luis Caneca dos Santos

Certamente, Future Knight já foi esquecido em muitas casas onde o MSX se encontra. Isto por se tratar de um jogo antigo e também muito difícil. Mesmo aqueles usuários que conseguiram vidas infinitas para ele, ou com um amigo ou por si mesmo, devem ter se complicado um pouco.

OBJETIVO DO JOGO

O objetivo do jogo é libertar a nossa amada princesa STALINA das terríveis garras do sono eterno. Mas para isso nosso querido personagem, o cavalheiro do futuro, terá que passar por 20 fases, repletas de inimigos, que tentarão de todas as formas diminuir sua quantidade de energia.

O JOGO

O jogo consta de 120 salas divididas em 20 fases e de vários objetos espalhados. Ele pode ser dividido em duas partes: a primeira, que se passa dentro da nave (que consta de 8 fases); e a segunda, que se passa fora da nave, constituída de 12 fases.

Infelizmente, não é possível o uso das setas para se controlar nosso personagem, como muitas pessoas estão acostumadas. Temos, com isto, que se contentar com o joystick ou com as letras 'Q", "W", "P", "L", que correspondem respectivamente à esquerda, direita, para cima e para baixo. Para acabar com nossos imimigos, usamos a barra de espaço, cuja função é soltar um dos três tipos de tiro existentes no jogo.

Nós possuímos três vidas que se esgotam quando nossa energia, que inicia em 999, chega a zero. É importante lembrar também que o cavalheiro não gosta de ficar fora de ação, isto é, imóvel, o que o deixa extremamente nervoso. Este nervosismo faz com que nosso amigo comece a pular, rodar, bater o pé e todos os tipos de rebeldia possível, o que faz você perder muita energia.

Outro ponto importante a ser esclarecido são as saídas trancadas. Estas saídas só podem ser abertas com um objeto específico, o qual devemos descobrir. Mesmo assim, não é necessário apenas pegar o objeto certo e ficar posicionado na saída fechada. Para acioná-lo, é necessário apertar as seguintes teclas simultaneamente: "V", "N" e "M". Existem apenas 3 destas mas, como o número de objetos é maior, tal tarefa torna-se difícil e monótona.

Para ajudá-lo a acabar este sensacional jogo, nós fornecemos o mapa das 8 primeiras fases (que constituem a parte difícil) e alguns macetes e ajudas para que tal tarefa citada acima não seja tão monótona e difícil.

MACETES

Aconselhamos que leiam esses macetes acompanhando visualmente o mapa publicado. Cada fase terá ao lado um número correspondente o qual será usado por nós nesta seção. E lá vão eles:

– Nós começamos o jogo, evidentemente, na fase 1. Nesta fase é necessário pegar o único objeto encontrado (SAFE PASS) e ir diretamente para a única saída encontrada.

– Na fase 2 devemos pegar a saída localizada na tela superior esquerda. Não mude de objeto nesta fase.

– Também na fase 3 não conveniente mudar de objeto. Devemos entrar na única saída existente.

– Na fase 4 troque o SAFE PASS pela PLATAFORM KEY e se direcione para a saída encontrada na tela superior direita.

– Na fase 5 apenas mude de objeto e volte para a fase 4.

– Com o SECURO KEY em nossas mãos podemos agora abrir a primeira porta trancada localizada na tela inferior esquerda apertando as três teclas mencionadas anteriormente. Com esta saída aberta voltamos uma tela, pegamos o SAFE PASS e entrarmos na saída que antes estava trancada.

– Na fase 7 devemos descer tudo e com o SAFE PASS abrir mais uma porta.

– Degemos, na fase 8, pegar o objeto (EXIT PASS) e voltar para a fase 2.

– Na fase 2 pega-se a saída localizada na tela inferior esquerda saindo, assim, na fase 6.

– Com o EXIT PASS em mãos podemos então abrir a última porta e finalmente sair da nave.

– Logo depois, em uma das fases seguintes, você encontrará um objeto denominado RELEASE SPELL. Pegue-o e não troque-o por nenhum outro. Com isto ao chegarmos a última tela do jogo nós encontraremos STALINA. Para quebrar o encanto devemos simplesmente voltar a tela anterior e apertar as teclas correspondentes para ativá-lo.

VIDA INFINITA

Para se conseguir vida infinita para o FUTURE KNIGHT é necessário que a linha do programa em BASIC onde está situado o último BLOAD fique assim:

```
BLOAD "FUTURE7.ASM" : POKE
&H8A39,0 : DEFUSR = PEEK (64704)
* 256 + PEEK(64703):A=USR(0)
```

É importante lembrar que não é necessário que sua última linha seja "FUTURE7.ASM".

SAÍDA 2
SAÍDA 5
SAÍDA 3
SAÍDA 3
SALA 2
SAÍDA 6
SAÍDA 1
SALA 4
SAÍDA 4
SALA 6

SAÍDA 4
SALA 3
SAÍDA 2

SAÍDA 5
SALA 5

SAÍDA 6
SAÍDA 7
SAÍDA 1
SALA 1
INÍCIO

TODAS AS SAÍDAS SÃO INTERLIGADAS ENTRE AS SALAS PELO NÚMERO DAS MESMAS. Ex: SAÍDA 3 NA SALA 2 COM SAÍDA 3 DA SALA 6

SAÍDA 7
SALA 7
SALA 8

VAMPIRE

- ○ LOCAL ONDE SE PEGA A CHAVE
- □ LOCAL ONDE SE DEIXA A CHAVE
- PERIGO
- PASSAGEM SECRETA
- CHAVE MESTRE
- BICICLETA
- JANELA
- ★ INICIO DO JOGO
- CRUZ (MATA O VAMPIRO)
- BIBLIA
- RODA
- ESTACA (MATA O VAMPIRO)
- MARTELO (MATA O VAMPIRO)
- SALA DUPLA

Pegue a chave 1 e deixe na porta 1.
Pegue a chave 2, abra a janela do quadro (7,6).
Pegue a chave 3, abra as janelas dos quadros (9,9), (9,8).
Abra a chave 3 (4,9).
Pegue a chave 4.
Pegue a cruz (5,5), abra a janela (5,6).
Abra a chave 4 e abra a janela (8,5).
Pegue a chave 5.
Abra a chave 2 (6,10).
Abra a chave 2 (4,11).
Pegue a (bíblia 6) (9,14).
Pegue a (roda 7) (5,13).
Abra a chave 6 (bíblia) (10,16).
Deixe a (roda 7) na bicicleta balançando o joystick ou batendo as duas setas para o lado rapidamente (14,13). Pegue a chave mestre.
Pegue o (martelo 8) (9,17).
Abra a chave mestre (7,5).
Pegue a (estaca 9) (4,16).
Vá à sala da chave mestre (7,5).
Abra a janela (7,4).
Vá no caixão, então se você tiver feito tudo o que eu disse, você lutará contra umas bolas e depois, uns corações.
Use o espaço para atirar.
E depois você verá o grande final.
Dica e macete: Alexandre de Oliveira Fernandes

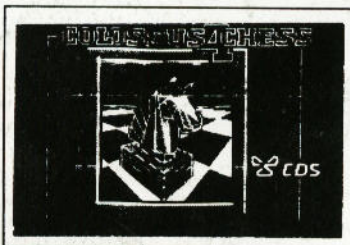
Imunidade

```
BLOAD "VAMP1", R
BLOAD "VAMP2", R
BLOAD "VAMP3", R
POKE&HA538, 201: POKE&HA578,201: POKE&HA56B, 201
DEFUSR = &H8590
A = USR0
```

MSX ESPETACULAR PROMOÇÃO MSX

SUPER PACOTE 1

SABRINA-
COLOSSUS
4 CHESS

NCz$ 15,00

SUPER PACOTE 2

TUAREG-
FIRE START-
LAST NINJA

NCz$ 15,00

SUPER PACOTE 3

MUTAN ZONE I-
PHARA'OS
REVENGE-
PAC MANIA

NCz$ 15,00

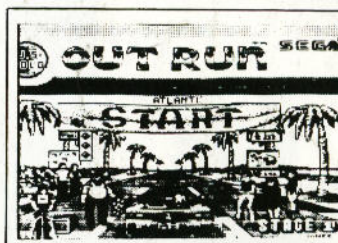
SUPER PACOTE 4

MUTAN ZONE II-
OUT RUN-
THE "A" TEAM

NCz$ 15,00

SUPER PACOTE 5

RALLY
PARIS DAKAR-
FLASH SPLASH
SEWER SAM

NCz$ 15,00

SUPER PACOTE 6

HYPER BALL-
WORM IN
PARADAISE-
TAWARA

NCz$ 15,00

SUPER PACOTE 7

OPERATION WOLF-
THE POWER
OF DARKNESS-
FINAL
COUNTDOWN

NCz$ 15,00

SUPER PACOTE 8

PLAY HOUSE-
SNAKE IT-
GUERRA DAS
FAMÍLIAS

NCz$ 15,00

SUPER PACOTE 9

THE FLINTSTONES-
ELITE-
KATAPU

NCz$ 15,00

× AIDS

— MUNDIALMENTE INÉDITO —

"AIDS" é um programa profissional, voltado para comunidade médico-científica (medicina diária – patologia clínica – instituições seguradoras previdenciária – medicina do trabalho e medicina legal) que possibilita realizar os seguintes cálculos:

- a data estimada do contágio inicial pelo HIV.
- a data provável de confirmação do diagnóstico.
- a data-limite de sobrevivência à doença.
- o tempo máximo de incubação do HIV.
- o tempo máximo de duração da doença propriamente dita.
- o tempo máximo entre o contágio inicial e o óbito.
- a estimativa do número absoluto de linfócitos do tipo T4–helper em qualquer data desejada, usando como dados necessários, duas contagens sucessivas do número absoluto dos linfócitos T-4 helper e suas respectivas datas.

AUTOR
DR. PIERRE CIRIADES
DIRETOR CLÍNICO DO HOSPITAL AMICO
(PROGRAMA REGISTRADO NA S.E.I.)

Envie seu pedido através de cheque nominal à NEWSOFT INFORMÁTICA LTDA.
Av. Nilo Peçanha, 50 sala 906 – CEP 20.020 – Rio de Janeiro–RJ – ou através de vale postal "AG. ARCOS cod. 522317